

第 9 講 データサイエンスにおけるプログラミング基礎

【学習到達目標】

- ・プログラミングの基本的な概念と構文を理解し、データ処理や分析に必要な基本操作を実行できる。
- ・代表的なプログラミング言語（例：Python）の基本的な文法とライブラリの使い方を習得し、簡単なデータ分析プログラムを作成できる。
- ・データの読み込み、加工、可視化といった基本的なプログラミングスキルを身につけ、データサイエンスの基礎的な作業を自律的に行える。

1. プログラミングの基本概念

(1)プログラミングとは

プログラミングとは、あらかじめ定めた計算やファイル操作などの手続きを計算機に処理させるための命令（プログラム）を作ることである。データサイエンスにおいては、データの収集、前処理、分析、可視化といった一連の作業を自動化し、効率的に行うためにプログラミングが必要となる。

プログラミングのために開発された言語をプログラミング言語という。プログラミング言語にはCやJavaなど様々な種類があるが、データサイエンス、特に機械学習分野においてはPythonという言語が良く用いられる。

(2)プログラミングの基本概念

プログラミングにおいて、具体的な値を変化させることができる存在を変数、変数の種類を型と呼ぶ。型の具体例としては、整数型(int 型)、浮動小数点型(float 型)のように数値が値となるものの他に、"apple"や"Hello World!"などの文字列型(str 型)や、[1, 3.14, "banana"]のように複数の要素を並べた配列型(list 型)などが存在する。これらの変数と、変数に対して四則演算などの演算を行う演算子や、条件分岐、ループ、関数の適用などを組み合わせることで様々な操作をコンピュータにさせることができるようになる。

統計学的な処理に対しては、R という言語が使われることも多い。

この解説では、公式版のPythonをコンソール(コマンドプロンプトやターミナル)上で動かすことを想定する。ダウンロードは公式サイト(<https://www.python.org/downloads/>)から。

2. Python と基本的な文法

以降は Python を例に、具体的なプログラミングの方法を解説する。

(1)変数への代入と基本操作

まずは変数に値を代入するプログラムを示す。

```
x = 1
y = "apple"
```

このプログラムは、「変数 x に 1 という値を代入し、変数 y に "apple" という値を代入せよ」という意味であり、実際に実行するとその通りに値が代入される。ここで注意すべきは、「 $=$ 」という記号は数学のように「両辺が等しい」という意味ではなく、「左辺の変数に右辺の値を代入する」という意味になる点である。なお、Python は変数に値を代入した時点で、その変数の型を自動的に判定する。尚、このプログラムは代入以外の操作をしないため、実行しても何も出力されない。

次に、以下のように変数の四則演算を行うプログラムを作ってみる。

```
a = 5
b = 2

print(a + b, a - b, a * b, a / b, a // b, a % b)
```

こちらは

```
7 3 10 2.5 2 1
```

という結果が得られる。各演算子の意味は以下の表の通りである。また、`print()` は「括弧の中身を表示せよ」という命令である。

二項演算子	意味
$a + b$	和
$a - b$	差
$a * b$	積
a / b	(実数としての)商
$a // b$	(a / b)を超えない最大の整数
$a \% b$	$a - b * (a // b)$

表 1 Python における四則演算の演算子

プログラム上では、文字列は引用符 (") または二重引用符 (") で囲うことに注意。

例えば、単に x と書いた場合「変数 x 」と解釈される一方、`'x'` または `"x"` と書くと「文字列 x 」と解釈される。

C や Java など、Python 以外の言語の場合、変数の型をあらかじめ宣言しておく必要があるものも存在する。

`print()` のように、特定の処理をまとめて行えるものを関数という。

ここで、「a // b」と「a % b」は整除法（余りが出る割り算）の商と剰余に近いものだが、Python の場合 a や b が float 型でも定義できる点に注意。

（他の言語では仕様が異なる例が多い。）四則演算以外で重要な演算子としては、「a ** b」（a の b 乗）などがある。

(2)条件分岐と繰り返し

条件分岐は if 文、繰り返しは for 文または while 文を使う。まずは if 文による条件分岐の例として、a に代入した数値（今回は 57）が 3 の倍数かどうかを判定するプログラムを考える。

```
a = 57

if a%3 == 0:
    print("a is a multiple of 3")
else:
    print("a is not a multiple of 3")
```

このプログラムにおいて、「a%3 == 0」は「a を 3 で割った余り(a%3)が 0 に等しい」という意味であり、その条件を満たす場合に行う命令「print("a is a multiple of 3")」は一段階インデントを下げた記述が必要がある。条件を満たさないときに別の命令を行いたい場合は、if 文終了後に「else:」と記述してから、同様にインデントを下げた条件を満たさない場合の命令を記述する。また、条件 A を満たさないときに、更に別の条件 B を満たしているか判別してから命令をしたい場合、「elif 条件 B:」とすればよい。

なお、どの場合においても、条件下での命令が複数行にわたる場合は、各行のインデントは揃える必要がある。

次に、繰り返しの例として 0 から 9 までの整数を足すプログラムを作る。

```
Sum = 0
for n in range(10):
    Sum += n

print(Sum)
```

これは以下のような結果になるはずである。

45

インデントは半角スペース何個分でも問題ないが、4 つにするのが主流である。

また、以下のように、for 文や if 文の中に、別の for 文や if 文を書くこともできる：

```
for n in range (10) :
    if n%2 == 0 :
        print( n, "is an even number.")
    else :
        print( n, "is an odd number.")
```

この際は、インデントを更に下げる必要があることに注意。

3. Python のライブラリ

プログラム開発者向けに便利なプログラムをまとめたものをライブラリという。Python のメリットは、以下の例のような多数のライブラリが開発されていることにある。

Numpy : 配列の操作、特に線形代数に関する演算を高速で行う

Pandas : データフレームの作成や欠損値処理など、データ管理において有用な機能が充実している

Scikit-learn : 各種の機械学習アルゴリズムを実行する

Matplotlib, Seaborn : データをグラフとして可視化する。

ここでは、例として Numpy を使うコードを紹介する。具体的には配列の要素の合計、平均値、中央値を計算する。

```
import numpy as np

a = np.array([ 1.0, 5.0, 3.0])
print( np.sum(a) ) # 配列の要素の和
print( np.mean(a) ) # 配列の要素の平均
print( np.median(a) ) # 配列の要素の中央値
```

このコードの1行目の「import numpy as np」という文は、「以降 numpy を np という名前で使う」という宣言である。その上で、numpy に含まれる関数「sum()」、「mean()」、「median()」をそれぞれ「np.sum()」、「np.mean()」、「np.median()」という名前で呼び出している。

4. 大規模言語モデル(LLM)の利用

Python を使いこなすには、Python 本体だけでなく、各種ライブラリの機能を把握する必要がある。従来はそれらを習得する際、教科書や Web 上の資料を調べる必要があったが、近年では ChatGPT や Copilot といった大規模言語モデル(LLM)の性能が向上し、自然言語による指示でもかなり正確なコードを書いてくれるようになった。そのため、特に使い慣れていないライブラリがある場合や、ライブラリがサポートしているかどうか分からない機能を使いたい場合などは、LLM に相談することも検討する価値がある。以下に ChatGPT o4-mini にコードを書いてもらった場合の例を挙げる。

ChatGPT o4-mini への指示 : Matplotlib で関数 $y = x^2$ のグラフを描くソースコードを作ってください。

ChatGPT o4-mini からの回答 : 以下のコード例で、Matplotlib を使って関数 $y = x^2$ のグラフを描画できます。

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

# x の値を -10 から 10 まで 0.1 刻みで生成
x = np.arange(-10, 10.1, 0.1)

# 関数 y = x^2 の計算
y = x**2

# グラフの描画
plt.figure(figsize=(8, 6))          # 図のサイズを設定
plt.plot(x, y, label=r'$y = x^2$') # ラベルに LaTeX 数式を使用
plt.title('Graph of $y = x^2$')    # タイトル
plt.xlabel('x')                    # x 軸ラベル
plt.ylabel('y')                    # y 軸ラベル
plt.grid(True)                     # グリッドを表示
plt.legend()                        # 凡例を表示

# グラフを画面に表示
plt.show()
```

(以下、コードの要点に関する説明があったが、省略)

このように、LLM はライブラリを使うようなコードでも容易に書いてくれる。ただし、現状では必ず正しいコードを出力してくれるとは限らないため、必ず動作確認をすること。特に、長く複雑なコードの場合はどこかで齟齬を発生させる危険性が増すため、そうした場合は想定している操作全体を幾つかの小さい部分に分け、別々にコードを作ってもらうのも良い。（例えば操作 A で計算した値を操作 B に処理させるならば、操作 A のコードと B のコードを別々に作ってもらうなど）また、漠然とした指示を出した場合 LLM 側が指示内容を誤解する恐れがあるため、「どのライブラリを使ってどんな計算をさせたいのか」などといった指示内容は、可能な限り具体的に書くのが望ましい。

参考文献

[1] 三谷純、「Python ゼロからはじめるプログラミング」（翔泳社、2021）

課題

1. Python を用いて、リストに格納された数値データの平均値と中央値を計算するプログラムを作成してください。
2. pandas ライブラリを使って、CSV ファイルからデータを読み込み、特定の列の欠損値を平均値で埋める処理を行うコードを書いてください。
3. matplotlib や seaborn を用いて、データの散布図とヒストグラムを作成し、データの分布や関係性を視覚的に表現してください。