

# 5. データマネジメント

<https://www.kkaneko.jp/cc/enshu2/index.html>

金子邦彦



- 
- ①Pandasデータフレームの実用性
  - ②外れ値の検出と処理
  - ③線形回帰とモデルの評価



# アウトライン

1. イントロダクション
2. AIの基礎
3. データマネジメント

# Google Colaboratory



The screenshot shows the Colaboratory homepage with a sidebar containing links like 'Colaboratory の概要', 'はじめに', 'その他のリソース', '機械学習の例: Seedbank', and 'セクション'.

The main area displays a code cell with the following Python code:

```
x = [5, 4, 1, 3, 2]
for i in x:
    print(i * 120)
```

The output of the code is:

```
600
480
120
360
240
```

The screenshot shows two code cells in the Colaboratory interface.

The first code cell contains:

```
[1]: x = 100
```

The second code cell contains:

```
[2]: if (x > 20):
    print("big")
else:
    print("small")
```

The output of the second cell is:

```
big
```

The third code cell contains:

```
[3]: s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)
```

The output of the third cell is:

```
15
```

URL: <https://colab.research.google.com/>

- オンラインで動く
- Python のノートブックの機能を持つ
- Python や種々の機能がインストール済み
- 本格的な利用には、Google アカウントが必要

# Google Colaboratory の全体画面



メニュー

(目次, 検索と置換,  
変数, ファイル)

The screenshot shows the Google Colaboratory interface. At the top is a toolbar with navigation icons (back, forward, search, etc.). Below it is a menu bar with 'Colab' and 'PRO' status, followed by 'Colab の定期購入を最大限に活用する'. The menu items are 'ファイル', '編集', '表示', '挿入', 'ランタイム', 'ツール', and 'ヘルプ'. A blue box highlights the 'メニュー' (Menu) item in the top right of the menu bar. To the left is a sidebar with icons for '目次' (Table of Contents), '検索' (Search), '変数' (Variables), and 'ファイル' (File). The main area contains three code cells:

- 1. 変数**

```
[2]: x = 100  
      y = 200
```
- 2. 式**

```
print(x + y)  
print(3 * x + y)  
300  
500
```
- 3. 条件分岐**

```
[4]: if (x > 50):  
      print('big')  
    else:  
      print('small')  
big
```

メニュー

コードセル, テキストセル  
の追加

コードセル,  
テキストセルの  
並び

Web ブラウザの画面

# Google Colaboratory のノートブック



## コードセル, テキストセルの2種類

- ・**コードセル** : Python プログラム, コマンド, 実行結果
- ・**テキストセル** : 説明文, 図

2. 式

← テキストセル

```
[5] print(x + y)
      print(3 * x + y)
```

← コードセル

300  
500

3. 条件分岐

← テキストセル

```
if (x > 50):
    print('big')
else:
    print('small')
```

← コードセル

big

# Google Colaboratory の本格的な機能 (使用には Google アカウントが必要)



- ・ノートブックの新規作成, 編集, 保存, 公開  
(Google Drive との連携による)
- ・公開により, 第三者がノートブックをダウンロードし, 編集や実行なども可能
- ・Python プログラム (コードセル内) の編集, 実行
- ・「!pip」や「%cd」などのシステム操作のためのコマンド (コードセル内) の編集, 実行
- ・ファイルのアップロード, ダウンロード
- ・ドキュメントの編集 (図, リンク, 添付ファイルを含めることができる)

## 5-1. イントロダクション

# Python の Pandas データフレーム

表形式のデータ

	x	y
0	1	4
1	1	2
2	1	5
3	2	4
4	3	5
5	3	3

データ本体

## 5-2. AI の基礎

# 機械学習と訓練データとプログラム



# 機械学習のプログラム

## 学習に使用する訓練データ (抜粋)

4	1	0	7	8	41078
1	2	7	1	6	12716
6	4	7	1	3	64773
3	7	9	9	1	37991
0	6	6	9	9	06699

画像 60000 枚  
(うち一部)

正解 60000個  
(うち一部)

## プログラム

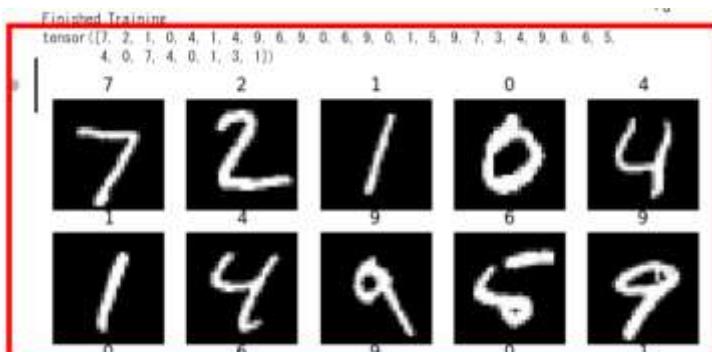
データを用いて学習を行う  
学習ののち、画像分類を行う

```

21 line = readLine("Please enter a file name: ");
22 if (line == null) {
23     System.out.println("No file name entered");
24     return;
25 }
26 String fileName = line;
27 String fileContent = null;
28 try {
29     fileContent = readFile(fileName);
30 } catch (IOException e) {
31     System.out.println("Error reading file " + fileName);
32 }
33 System.out.println(fileContent);
34
35 // 3. 从文件读取数据
36 String[] dataLines = fileContent.split("\n");
37 String[] data = new String[dataLines.length];
38 for (int i = 0; i < dataLines.length; i++) {
39     data[i] = dataLines[i];
40 }
41
42 // 4. 对文本进行处理
43 String[] words = new String[100];
44 int wordCount = 0;
45 for (String s : data) {
46     String[] tokens = s.split("\\s+");
47     for (String token : tokens) {
48         if (!token.equals("")) {
49             words[wordCount] = token;
50             wordCount++;
51         }
52     }
53 }
54
55 // 5. 将文本写入文件
56 String output = "";
57 for (String word : words) {
58     output += word + " ";
59 }
60
61 File outputfile = new File("output.txt");
62 try {
63     outputfile.createNewFile();
64     BufferedWriter bw = new BufferedWriter(new FileWriter(outputfile));
65     bw.write(output);
66     bw.close();
67 } catch (IOException e) {
68     System.out.println("Error writing file " + outputfile.getName());
69 }
70
71 System.out.println("Output file created successfully!");
72
73 // 6. 从文件读取数据
74 String[] dataLines2 = readFile("output.txt").split("\n");
75 String[] data2 = new String[dataLines2.length];
76 for (int i = 0; i < dataLines2.length; i++) {
77     data2[i] = dataLines2[i];
78 }
79
80 // 7. 对文本进行处理
81 String[] words2 = new String[100];
82 int wordCount2 = 0;
83 for (String s : data2) {
84     String[] tokens2 = s.split("\\s+");
85     for (String token2 : tokens2) {
86         if (!token2.equals("")) {
87             words2[wordCount2] = token2;
88             wordCount2++;
89         }
90     }
91 }
92
93 // 8. 将文本写入文件
94 String output2 = "";
95 for (String word2 : words2) {
96     output2 += word2 + " ";
97 }
98
99 File outputfile2 = new File("output2.txt");
100 try {
101     outputfile2.createNewFile();
102     BufferedWriter bw2 = new BufferedWriter(new FileWriter(outputfile2));
103     bw2.write(output2);
104     bw2.close();
105 } catch (IOException e) {
106     System.out.println("Error writing file " + outputfile2.getName());
107 }

```

学習の結果、文字認識の能力を獲得

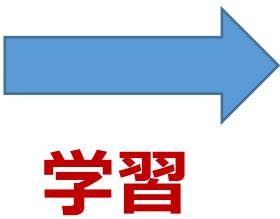
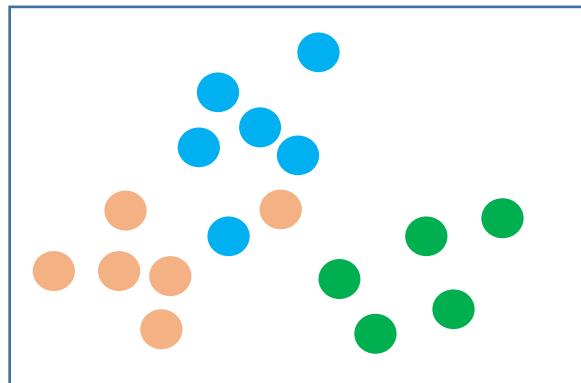




# 機械学習と訓練データ

機械学習は、コンピュータがデータを使用して学習することより知的能力を向上させる技術

## 訓練データ



学習者

3種類に分類済み

大量の訓練データを用いて  
学習を行う

# Iris データセットのロードと散布図 (Pandas データフレームを使用)

```
# 必要なライブラリをインポート
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import pandas as pd

# irisデータセットをロード (pandas を使用)
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)

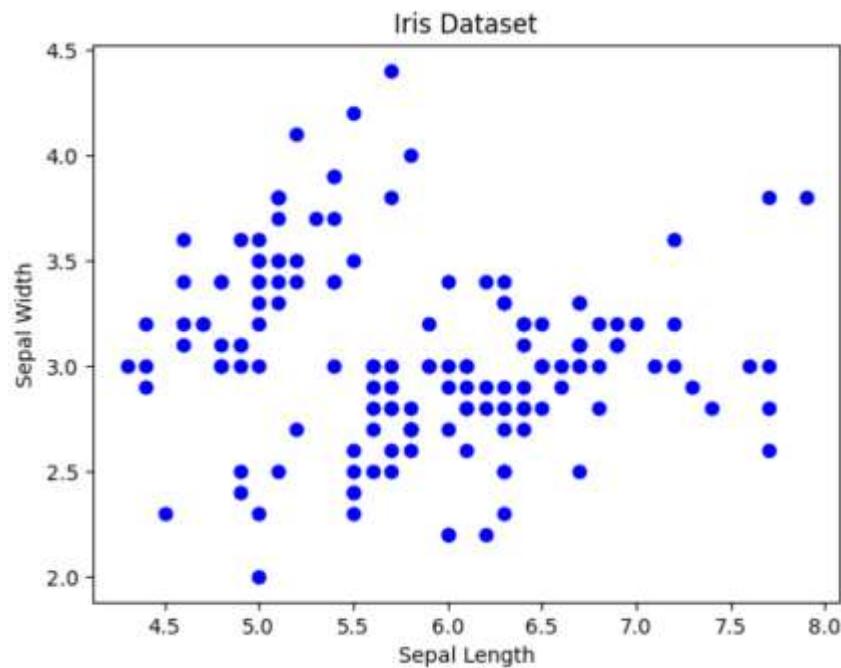
# データの先頭と末尾を確認
print("データの先頭:")
print(df.head())
print("データの末尾:")
print(df.tail())

# 必要な列だけを選択 (DataFrame形式でスライス)
X = df[['sepal length (cm)']]
y = df['sepal width (cm)']

# Matplotlibを用いて結果をプロット
plt.scatter(X, y, color='blue')

# 軸ラベルとタイトルを追加
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.title('Iris Dataset')

# グラフを表示
plt.show()
```



# 機械学習の例（線形回帰）



機械学習のうち 1 つ「線形回帰」を行う。線形回帰はデータに最もよく適合する線を見つけることである。

```
# 必要なライブラリをインポート
from sklearn.linear_model import LinearRegression
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import pandas as pd

# Irisデータセットをロード (pandas を使用)
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)

# データの先頭と末尾を確認
print("データの先頭:")
print(df.head())
print("データの末尾:")
print(df.tail())

# 必要な列だけを選択 (DataFrame形式でスライス)
X = df[['sepal length (cm)']]
y = df['sepal width (cm)']

# 線形回帰モデル
model = LinearRegression()

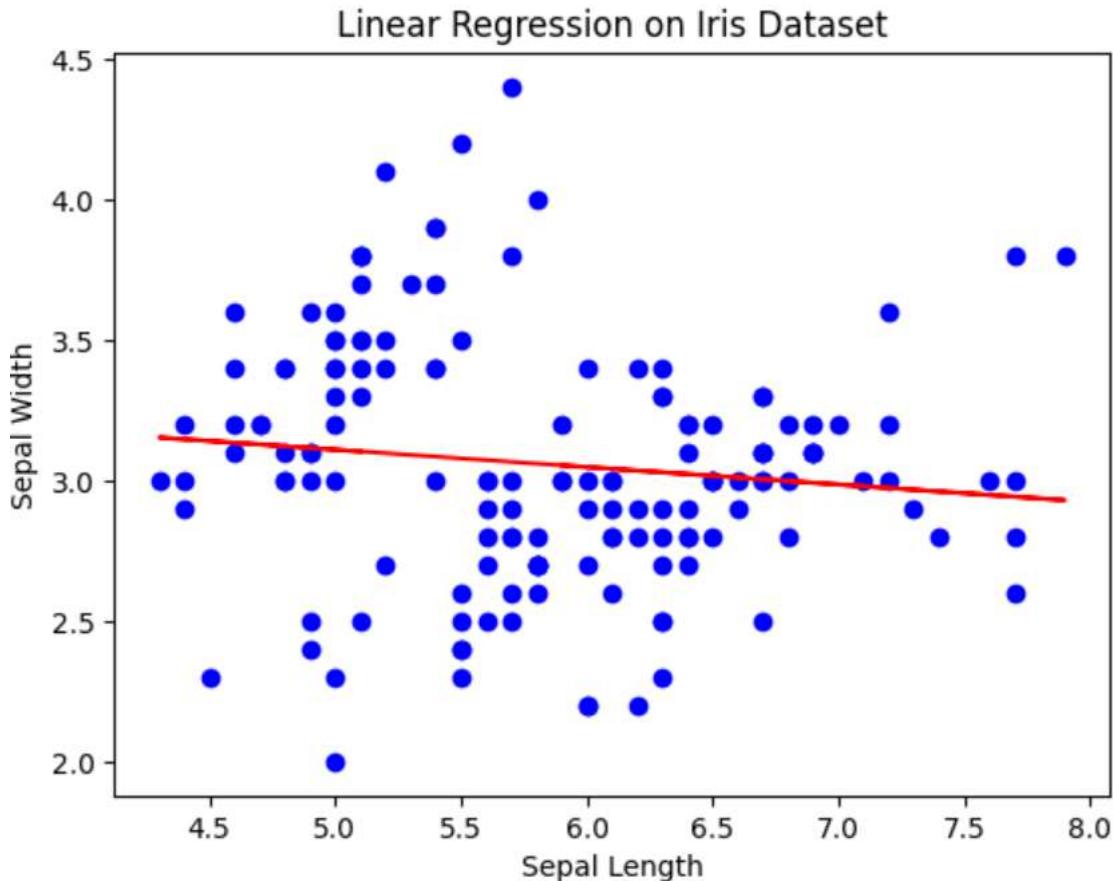
# 学習
model.fit(X, y)

# 予測を行う
y_pred = model.predict(X)

# Matplotlibを用いて結果をプロット
plt.scatter(X, y, color='blue')
plt.plot(X, y_pred, color='red')

# 軸ラベルとタイトルを追加
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.title('Linear Regression on Iris Dataset')

# グラフを表示
plt.show()
```



## 5-3. データマネジメント

# 外れ値



## 外れ値とは

- ・データ集合の中で、他のデータから**顕著に異なるデータ**
- 例: 年齢データにおいて、通常 0 から 120 程度の範囲が考えられる。1000 や -5 などの値が現れる場合は外れ値。

## 外れ値を取り扱うことのメリット

- ・**分析の精度向上**：平均、分散など統計的指標がより正確に反映されるようになる
- ・**機械学習モデルの性能向上**：外れ値を適切に処理することで、機械学習モデルが、外れ値にも適応するのを防ぐ

# 外れ値の主な検出方法と対処方法



## 外れ値の検出方法

- 統計的手法
  - 平均値からの偏差 (Zスコア)
  - データの分布を示す IQR (四分位範囲)
- 可視化による確認
  - 散布図、ボックスプロットを用いてデータの分布を視覚的に確認し、外れ値を特定

## 外れ値の対処方法

- 削除：外れ値を取り除く
- 置換：外れ値を中心値、平均値などの代表的な値で置換

# 欠損値と Python の NaN



## 欠損値

- ・データセット内で「**存在しない**」、「**測定されていない値**」、「**未知**」のものを欠損値という
- ・欠損値の原因: データ収集のミス、調査の未回答などが考えられる

## Python の NaN

- ・NaN は “Not a Number” の略語で、**「数値として定義されない値」** という意味
- ・Python では、欠損値や計算上のエラー（例：0での除算）を表現するために NaN を使用

# 線形回帰



- ・線形回帰はデータに最もよく適合する線を見つけることである。

例：家の大きさから、家の価格を予測

線形回帰モデル

$$\text{価格} = \beta_0 + \beta_1 \times \text{家の大きさ}$$

# Auto MPG データセット



車の性能や特性に関する情報を提供しており、これを基に燃料効率 (mpg) の予測モデルを構築することができる

- 作成: 1993年
- Python の CMU StatLib ライブラリ内
- 行数 398
- 属性
  - displacement: 排気量
  - mpg: 燃料消費 (ガロンあたりのマイル数)
  - cylinders: シリンダー数
  - horsepower: 馬力 (欠損値あり)
  - weight: 重量
  - acceleration: 加速
  - model\_year: モデル年
  - origin: 産地または原点
  - car\_name: 車名
- mpg 以外の全属性を「特徴」
- mpg を「ターゲット」



# 演習 データマネジメントの プラクティス

## トピックス】

- Pandas データフレーム
- 欠損値の処理
- 外れ値の処理
- データの分割（訓練データ、テストデータへの分割）
- 線形回帰モデルの評価

# データマネジメントのプラクティス



- データセットの確認
  - データの先頭と末尾を表示して確認
  - 平均、分散、標準偏差の確認
- 欠損値の処理
  - ここでは、欠損値を含む行は除去している。
- 外れ値の処理
  - 平均値からの偏差（Zスコア）を使用して外れ値を検出
  - ここでは、外れ値を含む行は除去している。
- データの分割
  - 訓練データとテストデータに分ける。
- モデルの学習と評価
  - 訓練データで線形回帰モデルを学習し、テストデータを用いて「うまく学習できたか」を評価。

## ①Pandasデータフレームの実用性

Pandasのデータフレームは、データ分析のための強力なツールとして知られており、実世界のデータの操作や解析に役立ちます。データフレームには、様々なデータ操作や分析を支援する機能が組み込まれており、これを習得することで、多様な課題への取り組みが可能になります。

## ②外れ値の検出と処理

データサイエンスや機械学習の分野では、外れ値が分析結果に大きな影響を及ぼすことがあります。外れ値の正確な検出と適切な処理方法を学ぶことは、データの質を高め、より正確な予測や分析につながります。

## ③線形回帰とモデルの評価

線形回帰は、データの予測に関する基本的な手法の一つであり、特徴とターゲットの関係を明確にする上で重要です。モデルの正確さを評価する際、まず、データを訓練データとテストデータに分けることが必要です。そして、訓練データで学習したモデルの性能は、必ず、テストデータを用いて評価されるべきです。