5．データマネジメント

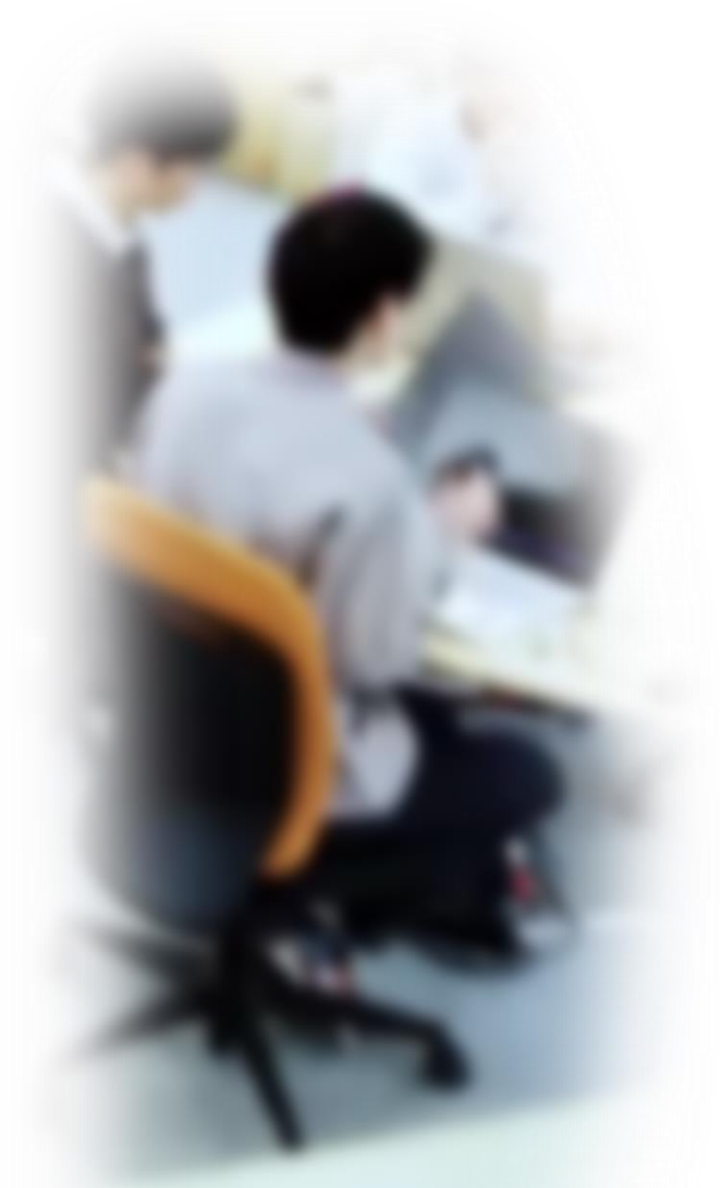
https://www.kkaneko.jp/cc/enshu2/index.html

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

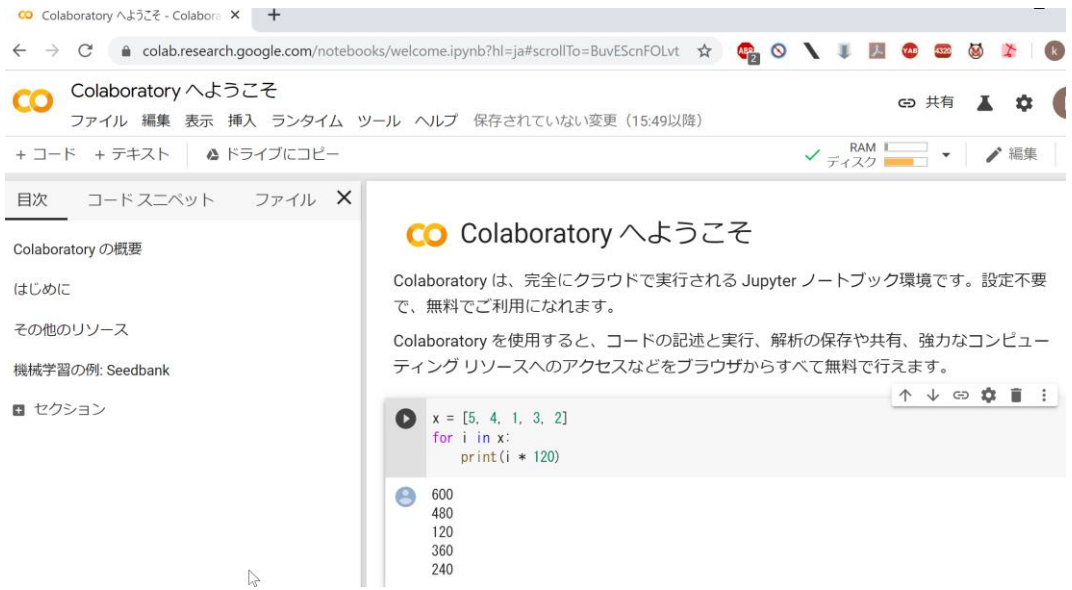
③線形回帰とモデルの評価

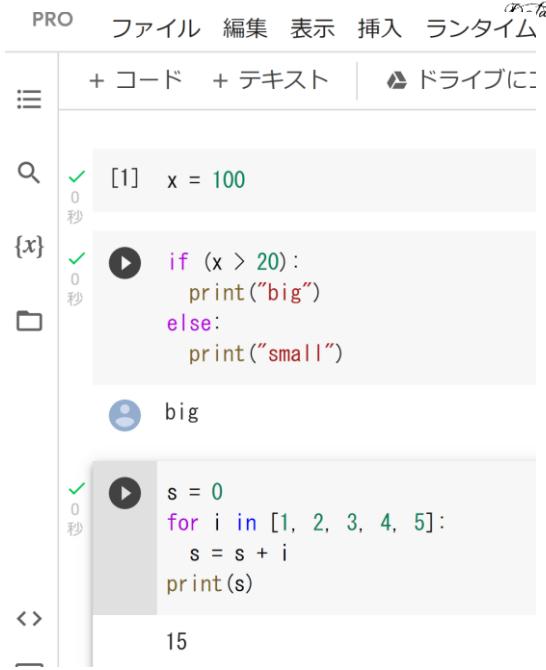


アウトライン

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

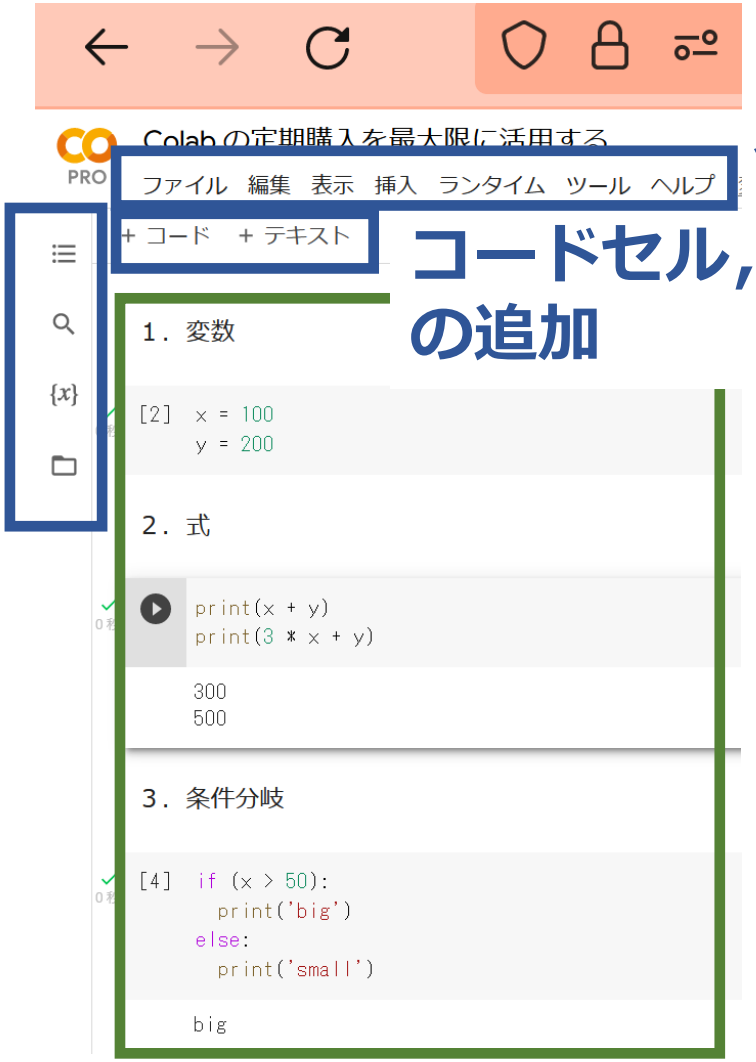




URL: https://colab.research.google.com/

- **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

コードセル，テキストセルの追加

メニュー
（目次，検索と置換，変数，ファイル）

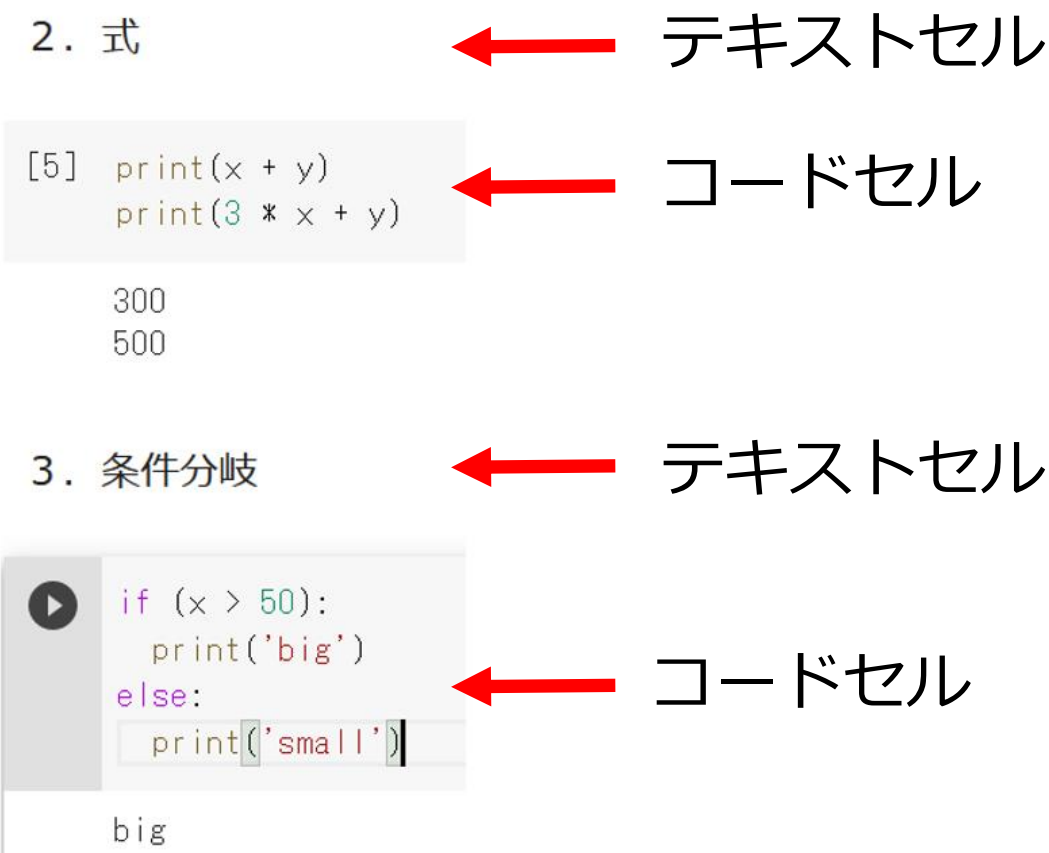
コードセル，テキストセルの並び

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



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

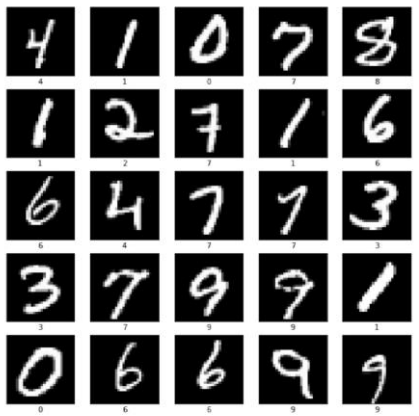
データ本体

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
1 2 7 1 6
6 4 7 7 3
3 7 9 9 1
0 6 6 9 9

画像 60000枚
（うち一部）

正解 60000個
（うち一部）

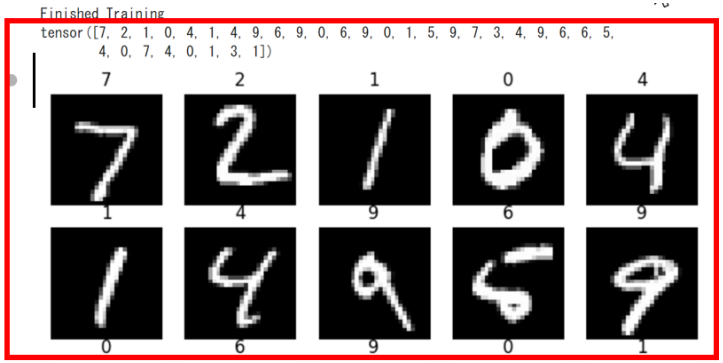
プログラム

データを用いて学習を行う
学習ののち，画像分類を行う

```
[4] !pip install -U scikit-learn matplotlib

    import torch
    import torch.nn as nn
    import torch.optim as optim
    from sklearn import datasets
    from sklearn.model_selection import train_test_split
    from sklearn.preprocessing import StandardScaler
    import matplotlib.pyplot as plt

    # データの取得と前処理
    iris = datasets.load_iris()
    X = iris.data
    y = iris.target

    # データの標準化
    scaler = StandardScaler()
    X = scaler.fit_transform(X)

    # 訓練データとテストデータの分割
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    X_train = torch.tensor(X_train, dtype=torch.float32)
    y_train = torch.tensor(y_train, dtype=torch.long)
    X_test = torch.tensor(X_test, dtype=torch.float32)
    y_test = torch.tensor(y_test, dtype=torch.long)

    # ニューラルネットワークの定義
    class Net(nn.Module):
        def __init__(self):
            super(Net, self).__init__()
            self.fc1 = nn.Linear(4, 10)  # 入力は4次元（Irisの特徴量）
            self.fc2 = nn.Linear(10, 3)  # 出力は3クラス

        def forward(self, x):
            x = torch.relu(self.fc1(x))
            x = self.fc2(x)
            return x

    net = Net()
    criterion = nn.CrossEntropyLoss()
    optimizer = optim.SGD(net.parameters(), lr=0.01)
```

学習の結果、文字認識の能力を獲得

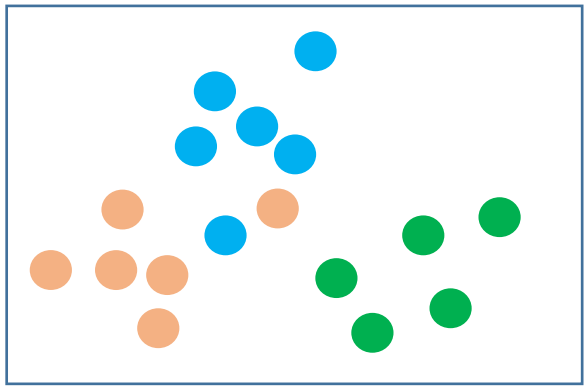
```
Finished Training
tensor([7, 2, 1, 0, 4, 1, 4, 9, 6, 9, 0, 6, 9, 0, 1, 5, 9, 7, 3, 4, 9, 6, 6, 5,
        4, 0, 7, 4, 0, 1, 3, 1])
```



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



3 種類に分類済み

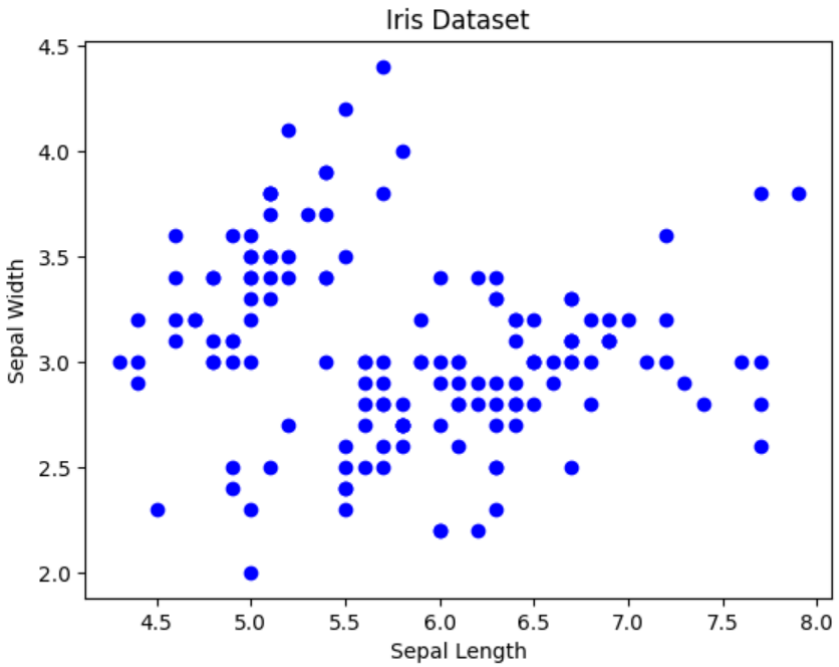
学習

学習者

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）

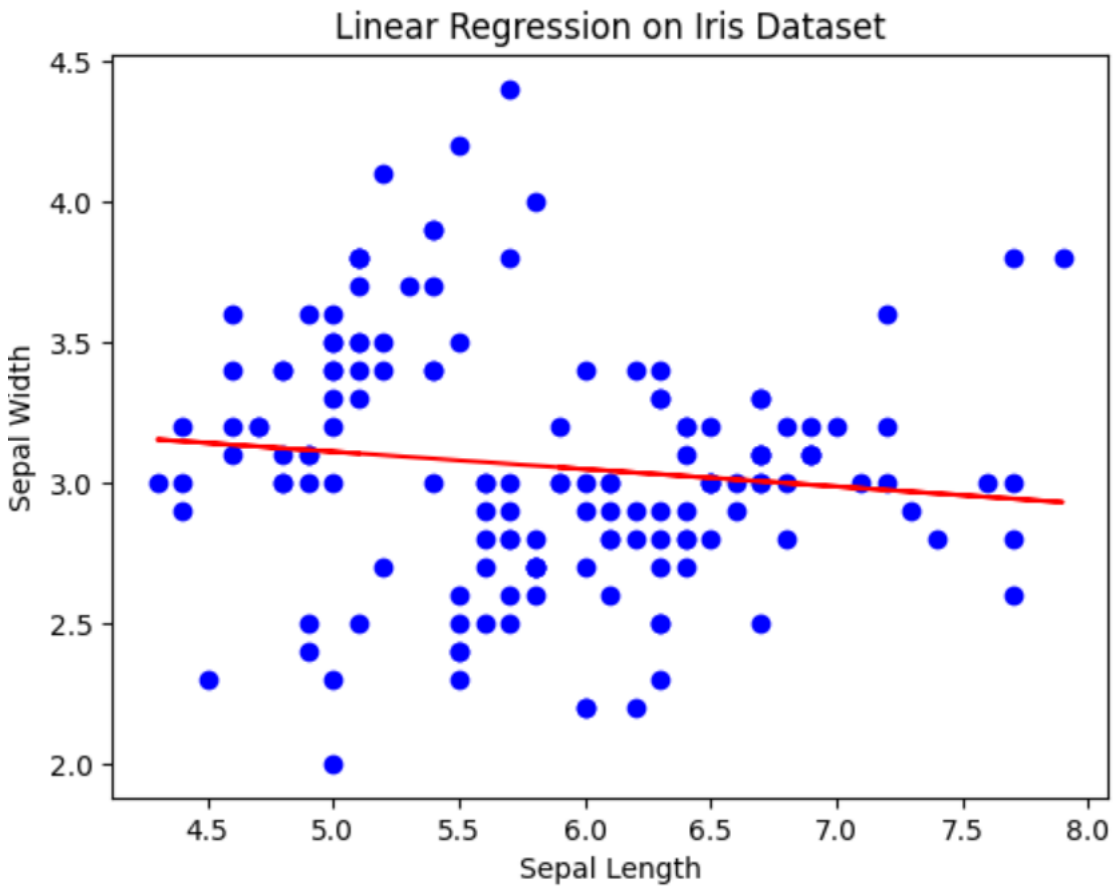
```python
# 必要なライブラリをインポート
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import pandas as pd

# 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。

```python
# 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：外れ値を中央値、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

価格 = β0 + β1 × 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**

① Google Colaboratory のページを開く

https://colab.research.google.com/drive/1oUtNxZsm81Bwm2dxblRF2a51bhkElFNz?usp=sharing

② プログラムや説明や実行結果が表示されるので確認

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。