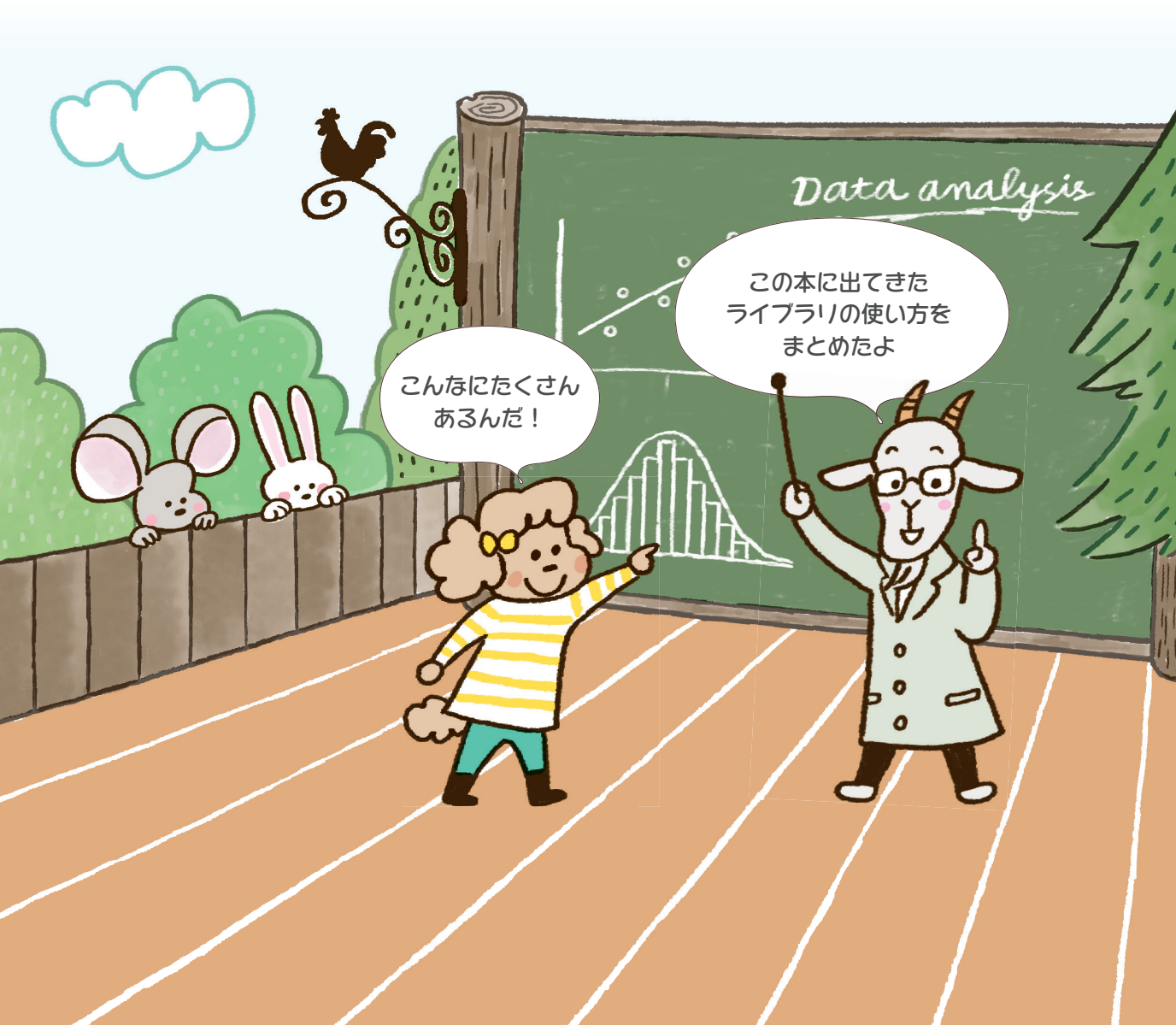
特典PDF
ライブラリ簡易マニュアル



pandas（パンダス）

表データの読み書きや集計、分析を行えるライブラリ

インポートする
`import pandas as pd`

行データから、データフレームを作る
`df = pd.DataFrame(data)`
CSVファイル（UTF-8形式）を読み込む
`df = pd.read_csv("ファイル名.csv")`
CSVファイル（Shift-JIS形式）を読み込む
`df = pd.read_csv("ファイル名.csv", encoding="Shift_JIS")`
CSVファイル（0列目がインデックス）を読み込む
`df = pd.read_csv("ファイル名.csv", index_col=0)`
CSVファイル（ヘッダーがない）を読み込む
`df = pd.read_csv("ファイル名.csv", header=None)`

列データを取り出す
`df["列名"]`
複数の列データを取り出す
`df[["列名", "列名"]]`
行データを取り出す
`df.iloc[[行番号]]`
複数の行データを取り出す
`df.iloc[[行番号, 行番号]]`
要素データを取り出す
`df.iloc[行番号]["列名"]`

空のデータフレームを作る
`df = pd.DataFrame()`
列データを追加する
`df["新列名"] = 列データ`
行データを追加する
`df = df.concat([データフレーム, データフレーム])`
列データを削除する
`df = df.drop("列名", axis=1)`
行データを削除する
`df = df.drop("行名")`

条件でデータを抽出
`df = df[条件]`

欠損値の個数
`df.isnull().sum()`
欠損値がひとつである行を削除
`df = df.dropna()`
指定した列で欠損値がある行を削除
`df = df.dropna(subset=["列名"])`
欠損値を平均値で埋める
`df = df.fillna(df.mean())`
欠損値をひとつ前の値で埋める
`df = df.ffill()`
欠損値を次の行の値で埋める
`df = df.bfill()`
重複データの個数
`df.duplicated().value_counts()`

重複データの２つ目以降を削除する
`df.drop_duplicates()`
文字列の列データを整数に変換する
`df["列名"] = df["列名"].astype(int)`
カンマ付き文字列の列データのカンマを削除する
`df["列名"] = df["列名"].str.replace(",","")`

列データの平均値を求める
`df["列名"].mean()`
各列データの平均値を求める
`df.mean()`
各列データの中央値を求める
`df.median()`
各列データの最頻値を求める
`df.mode()`
列データの度数分布表を表示する
`cut = pd.cut(df["列名"], bins=区切る範囲, right=False)` `cut.value_counts(sort=False)`

NumPy（ナンパイ）

数値計算や配列計算を効率的に行えるライブラリ

インポートする
`import numpy as np`

かたよりのないランダムな値を作る
`np.random.randint(最小値, 最大値, 個数)`
正規分布になるようなランダムな値を作る
`np.random.normal(平均値, 標準偏差, 個数)`

matplotlib（マットプロットリブ）

いろいろな種類のグラフを描画できるライブラリ

インポートする
`import matplotlib.pyplot as plt`

列データのヒストグラムを表示する
`df["列名"].plot.hist(bins=区切る範囲)`
列データの棒グラフを表示する
`df.plot.bar()`
列データの折れ線グラフを表示する
`df["列名"].plot()`
列データの円グラフを表示する
`df["列名"].plot.pie()`
列データの円グラフを表示する（一般的な円グラフ）
`df["列名"].plot.pie(startangle=90, counterclock=False, labeldistance=0.5)`
散布図を表示する
`df.plot.scatter(x="横軸の列名", y="縦軸の列名", color="色")`

タイトルを表示する

```
plt.title("タイトル")
```

凡例の位置を指定する

```
plt.legend(loc="指定文字列")
```

表示位置	指定文字列	数値
ベストな位置	best	0
右上	upper right	1
左上	upper left	2
左下	lower left	3
右下	lower right	4
右	right	5
中央左	center left	6
中央右	lower center	7
中央下	lower center	8
中央上	upper center	9
ど真ん中	center	10

横軸の目盛りを置き換える

```
plt.xticks(置き換える目盛りのリスト, 目盛りのリスト)
```

縦軸の目盛りを置き換える

```
plt.yticks(置き換える目盛りのリスト, 目盛りのリスト)
```

横軸の最大値、最小値を指定する

```
plt.xlim([最小値, 最大値])
```

縦軸の最大値、最小値を指定する

```
plt.ylim([最小値, 最大値])
```

seaborn（シーボーン）

きれいなグラフが描けて、データ分析に便利な機能がついているライブラリ

インポートする

```
import seaborn as sns
```

matplotlibを使う前に指定する

とりあえず、きれいなグラフにする

```
sns.set()
```

フォントを指定してきれいなグラフにする

```
sns.set(font=["フォント名"])
```

日本語フォントを指定してきれいなグラフにする（お使いの環境に合わせて先頭の「#」を削除して、コメントを解除してお使いください）

```
#sns.set(font=["Meiryo"]) # Windows
#sns.set(font=["Hiragino Maru Gothic Pro"]) # macOS
#sns.set(font=["IPAexGothic"]) # Colab Notebook
```

スタイルを指定してきれいなグラフにする（お使いの環境に合わせて先頭の「#」を削除して、コメントを解除してお使いください）

```
#sns.set(style="スタイル", font=["Meiryo"])   # Windows
#sns.set(style="スタイル", font=["Hiragino Maru  Gothic Pro"]) # macOS
#sns.set(style="スタイル", font=["IPAexGothic"])  # Colab Notebook
```

スタイル名	表示されるグラフ
dark	暗い色
darkgrid	暗い色で線あり
white	白
whitegrid	白で線あり
ticks	目盛り付き

列データの箱ひげ図を表示する
sns.boxplot(data=df, width=幅)
列データのヒストグラムを表示する（カーネル密度推定付き）
sns.histplot(df["列名"], kde=True, color="色", alpha=透明度, stat="density")
散布図 ＋ 回帰直線を表示する
sns.regplot(data=df, x="横列", y="縦列", line_kws={"color":"色"})
ヒストグラム付き散布図 ＋ 回帰直線を表示する
sns.jointplot(data=df, x="横の列名", y="縦の列名", kind="reg", line_kws={"color":"色"})

相関行列を色分けして表示する
sns.heatmap(df.corr())
散布図のすべての組み合わせを行列で表示する
sns.pairplot(data=df)
アヤメの品種データを読み込む
sns.load_dataset("iris")

scipy（サイパイ）

高度な科学計算を行えるライブラリ

インポートする
`from scipy.stats import norm`

正規分布の累積分布関数：ある値は、下からどんな確率（0〜1）なのか
`cdf = norm.cdf(x=調べたい値, loc=平均値, scale=標準偏差)`
正規分布のパーセント点関数：下からのある確率（0〜1）の値はなにか
`ppf = norm.ppf(q=パーセント値, loc=平均値, scale=標準偏差)`