

GoogleColaboratory動作確認手順

本手順書ではGoogleColaboratory動作確認手順について記述する。

GoogleColaboratoryとはGoogle社が提供している、ブラウザから直接Pythonを記述、実行できるサービス。

演習1-1 ～ 演習1-3 ではGoogleColaboratoryを使用して各種プログラムを実行する為、その流れを説明する。

目次

[1 演習用コンテンツコピー先の確認](#)

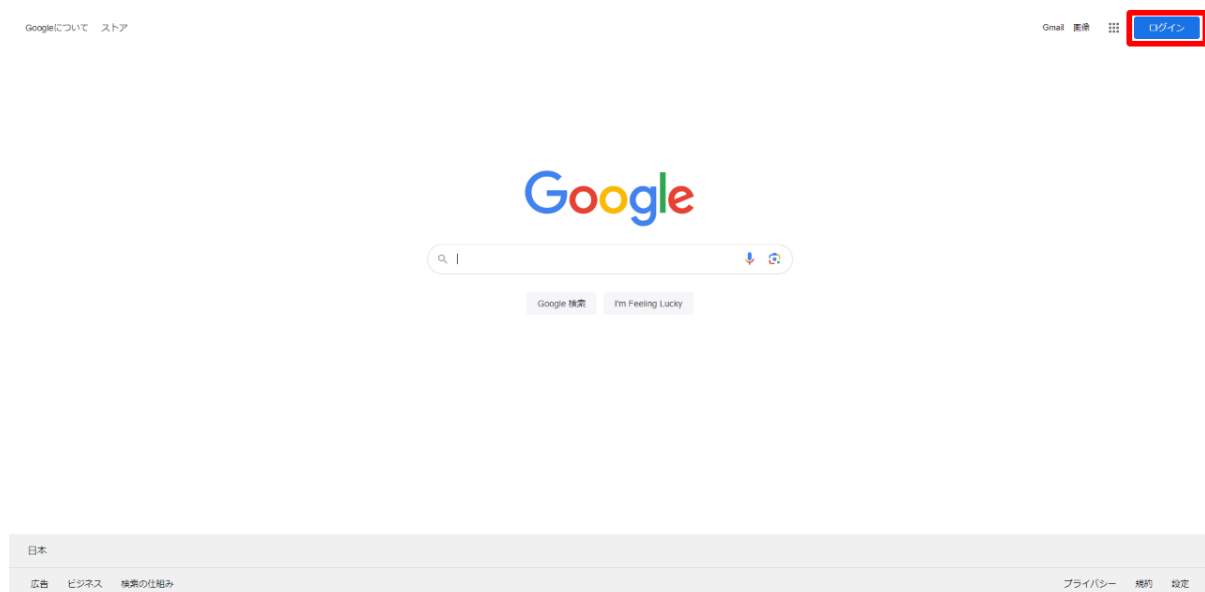
[2 演習用コンテンツのコピー](#)

[3 動作確認の実施](#)

1 演習用コンテンツコピー先の確認

事前にファイルのコピー先を確認する為、Googleドライブを開く

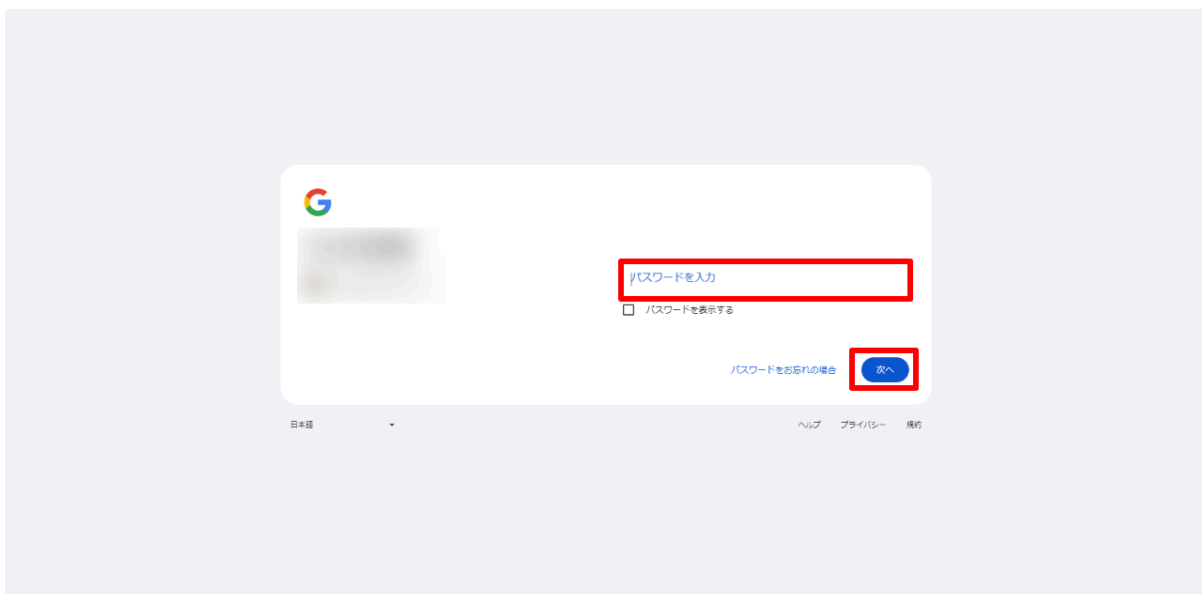
1. [Googleトップ画面](#)でログインボタンをクリック



2. 対象のGoogleアカウントを選択



3. パスワードを入力して「次へ」をクリック



4. ログインが成功するとGoogleトップ画面が表示される



5. 右上のアイコンを選択し、「ドライブ」をクリック



6. Googleドライブが表示されるので「マイドライブ」をクリック

※後続の手順実施後にこの場所に演習用コンテンツがコピーされる



2 演習用コンテンツのコピー

1. [演習1-1 のコンテンツ](#)に移動



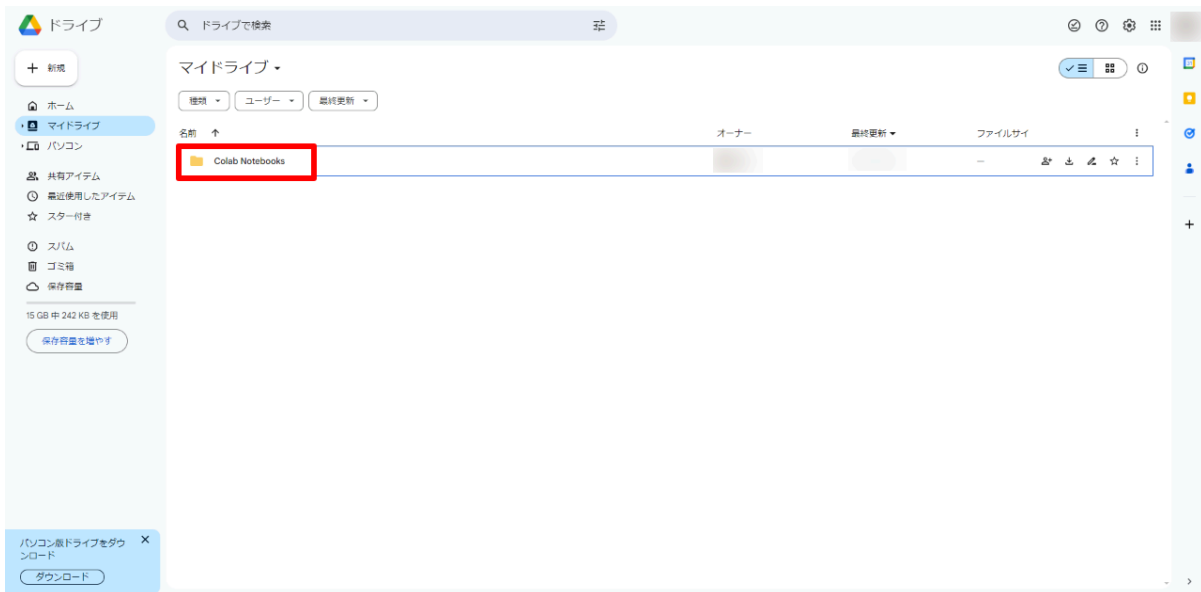
2. 「ドライブにコピー」をクリックし、自分のGoogleドライブにコピー ※東北大学寄附講座_演習1_1.ipynb のコピー というファイルが生成される



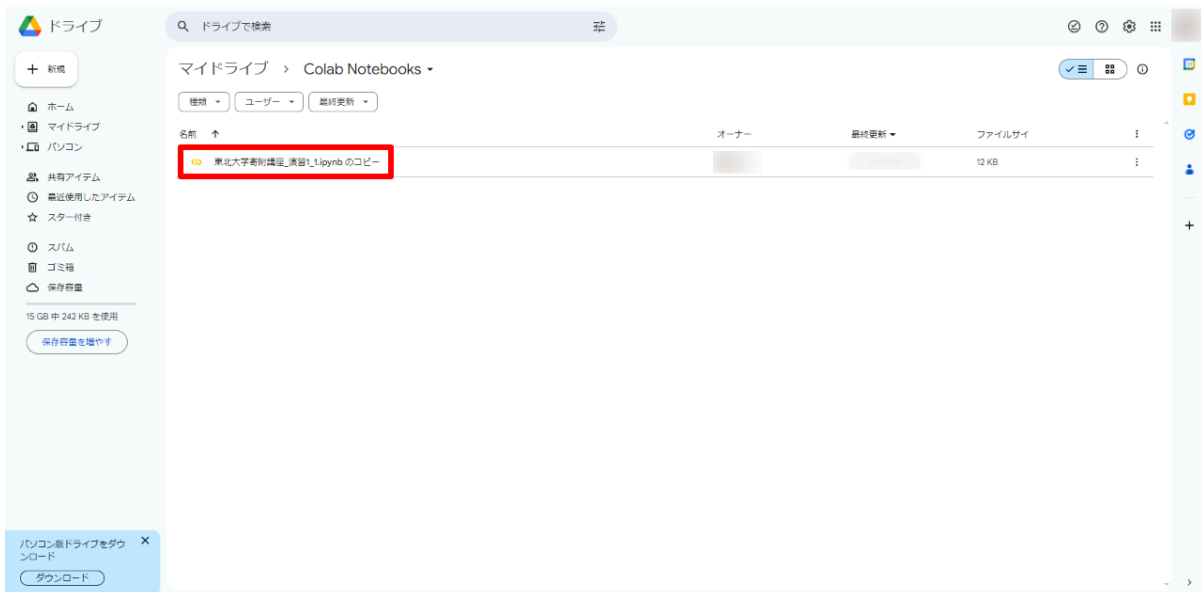
3. コピーが完了すると「東北大学寄附講座_演習1_1.ipynb のコピー」が開かれる



4. Googleドライブのマイドライブ上に「Colab Notebooks」フォルダが作成されていることを確認



5. Colab Notebooksフォルダ内に「東北大学寄附講座_演習1_1.ipynb のコピー」が存在することを確認



6. 演習1-2 のコンテンツに移動

東北大学 寄附講座 演習1: Pythonを用いたデータ分析・機械学習 (2)

データセットの読み込み

教師なし学習: k-means法を用いたクラスタリング

k-means法によるクラスタリングの実施

クラスタリング結果を可視化

```
[ ] import pandas as pd
from sklearn import datasets

# Iris データセットを読み込み
iris = pd.DataFrame(datasets.load_iris().data, columns=datasets.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print(f"データセットのサンプル: {iris.sample(10)}")
```

教師なし学習: k-means法を用いたクラスタリング

k-means法によるクラスタリングの実施

```
[ ] from sklearn.cluster import KMeans

# KMeans の初期化
kmeans = KMeans(n_clusters=3, random_state=0)

# 学習
kmeans.fit(iris)

# 予測
predict = kmeans.predict(iris)

print(f"各データが所属するクラスは以下と予測されたよ (predict)")
```

クラスタリング結果を可視化

7. 「ドライブにコピー」をクリックし、自分のGoogleドライブにコピー ※東北大学寄附講座_演習1_2.ipynb のコピー というファイルが生成される

東北大学 寄附講座 演習1: Pythonを用いたデータ分析・機械学習 (2)

データセットの読み込み

教師なし学習: k-means法を用いたクラスタリング

k-means法によるクラスタリングの実施

クラスタリング結果を可視化

```
[ ] import pandas as pd
from sklearn import datasets

# Iris データセットを読み込み
iris = pd.DataFrame(datasets.load_iris().data, columns=datasets.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print(f"データセットのサンプル: {iris.sample(10)}")
```

教師なし学習: k-means法を用いたクラスタリング

k-means法によるクラスタリングの実施

```
[ ] from sklearn.cluster import KMeans

# KMeans の初期化
kmeans = KMeans(n_clusters=3, random_state=0)

# 学習
kmeans.fit(iris)

# 予測
predict = kmeans.predict(iris)

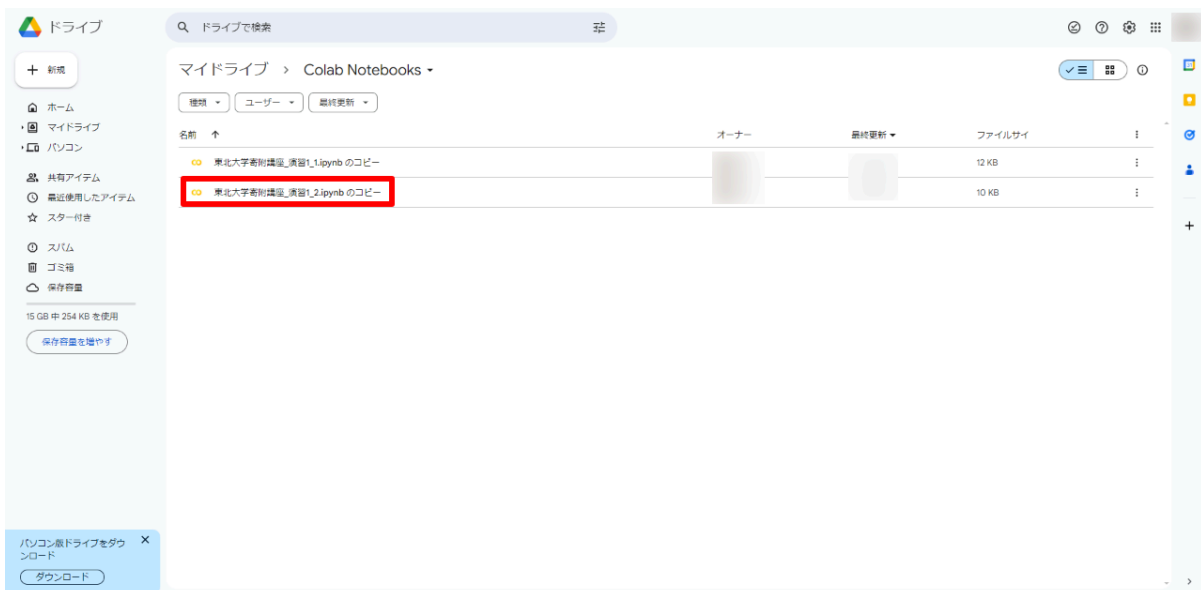
print(f"各データが所属するクラスは以下と予測されたよ (predict)")
```

クラスタリング結果を可視化

8. コピーが完了すると「東北大学寄附講座_演習1_2.ipynb のコピー」が開かれる



9. 4で作成された「Colab Notebooks」内に「東北大学寄附講座_演習1_2.ipynb のコピー」が存在することを確認



10. [演習1-3 のコンテンツ](#)に移動



東北大学 寄附講座 演習1: Pythonを用いたデータ分析・機械学習 (3)

ChatGPTを用いた自然言語対話とプロンプトエンジニアリング

ChatGPTを用いる準備

ライブラリのインストール

```
[ ] !pip install openai==1.3.5
!pip install langchain==0.0.340
!pip install langchain-experimental==0.0.42
!pip install tabulate==0.9.0
```

OpenAIのAPIキーを環境変数としてセット

```
[ ] !export os
os.environ["OPENAI_API_KEY"] = "<your openai api key>" # 実際は学生に自分のAPIキーへ書き換えてもらう
```

データセットの読み込み

```
[ ] !import pandas as pd
from sklearn import datasets

ds = datasets.load_iris()
iris = pd.DataFrame(ds.data, columns=ds.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print("データセットのサンプル")
iris.sample(10)
```

11. 「ドライブにコピー」をクリックし、自分のGoogleドライブにコピー ※東北大学寄附講座_演習1_3.ipynb のコピー というファイルが生成される



東北大学 寄附講座 演習1: Pythonを用いたデータ分析・機械学習 (3)

ChatGPTを用いた自然言語対話とプロンプトエンジニアリング

ChatGPTを用いる準備

ライブラリのインストール

```
[ ] !pip install openai==1.3.5
!pip install langchain==0.0.340
!pip install langchain-experimental==0.0.42
!pip install tabulate==0.9.0
```

OpenAIのAPIキーを環境変数としてセット

```
[ ] !export os
os.environ["OPENAI_API_KEY"] = "<your openai api key>" # 実際は学生に自分のAPIキーへ書き換えてもらう
```

データセットの読み込み

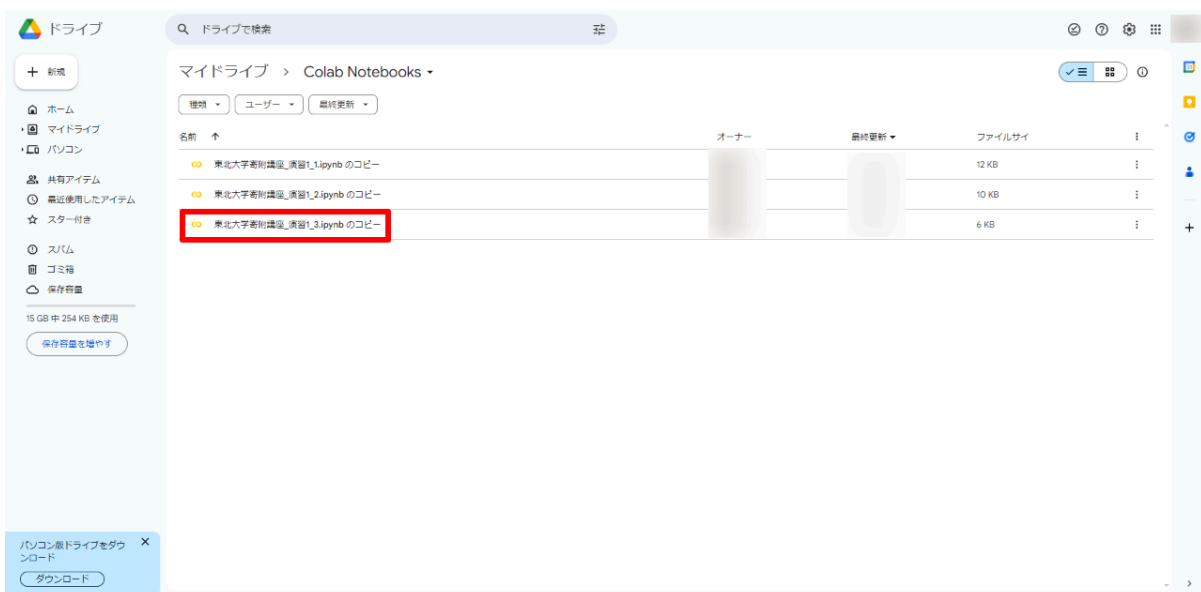
```
[ ] !import pandas as pd
from sklearn import datasets

ds = datasets.load_iris()
iris = pd.DataFrame(ds.data, columns=ds.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print("データセットのサンプル")
iris.sample(10)
```

12. コピーが完了すると「東北大学寄附講座_演習1_3.ipynb のコピー」が開かれる

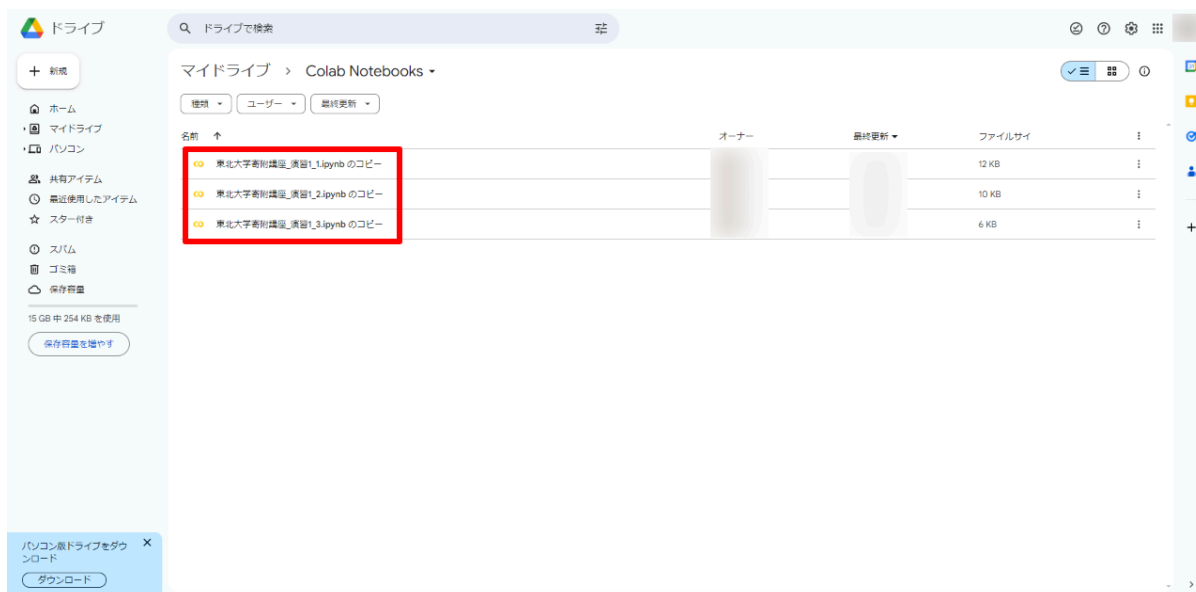


13. 4で作成された「Colab Notebooks」内に「東北大学寄附講座_演習1_3.ipynb のコピー」が存在することを確認



14. 4で作成された「Colab Notebooks」内にコピーした3ファイルが存在することを確認

※ここから使用するファイルは必ずコピーした3ファイルを使用すること

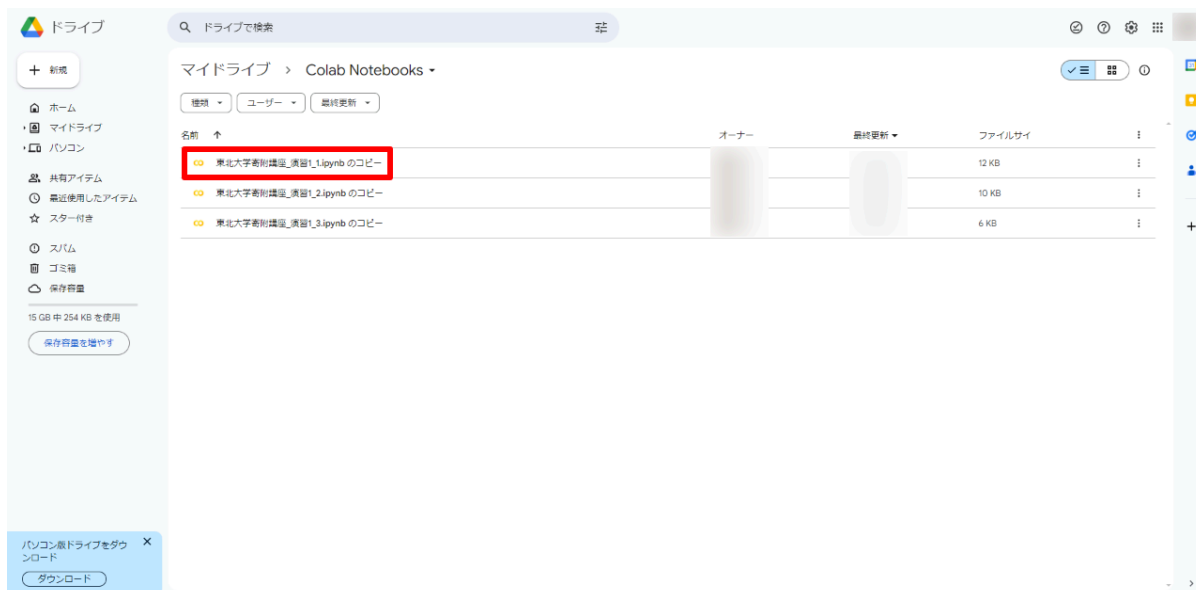


3 動作確認の実施

演習1-1

1. 東北大学寄附講座_演習1_1.ipynb のコピー を開く

※東北大学寄附講座_演習1_1.ipynbが開かれていない場合は、Googleドライブのマイドライブ内、「Colab Notebooks」フォルダから開く



(画面構成)

GoogleColaboratoryの画面構成を以下に記載する



①プログラム記載セル（グレーで塗りつぶされたセル）

②プログラム実行ボタン

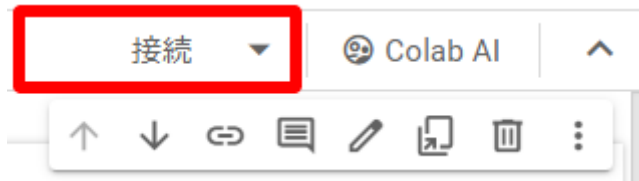
③説明

2. プログラム実行

プログラムが記載されているセルを確認し、左側のプログラム実行ボタンをクリック



※初回実行時は少し時間がかかるので、画面右上のステータスが「RAMディスク」になるまで待機



1. 「接続」 → プログラム実行前
2. 「接続中」 → プログラム実行ボタンをクリック
3. 「接続済」 → プログラム実行準備完了
4. 「RAMディスク」 → プログラム実行開始

3. 実行結果の確認

プログラムが記載されたセルの下に、実行結果が表示されるので確認

The screenshot shows a Jupyter Notebook interface in Colab AI. The notebook is titled 'Iris Versicolor', 'Iris Setosa', and 'Iris Virginica'. The code cell contains the following Python code:

```
import pandas as pd
from sklearn import datasets
import seaborn as sns
import matplotlib.pyplot as plt

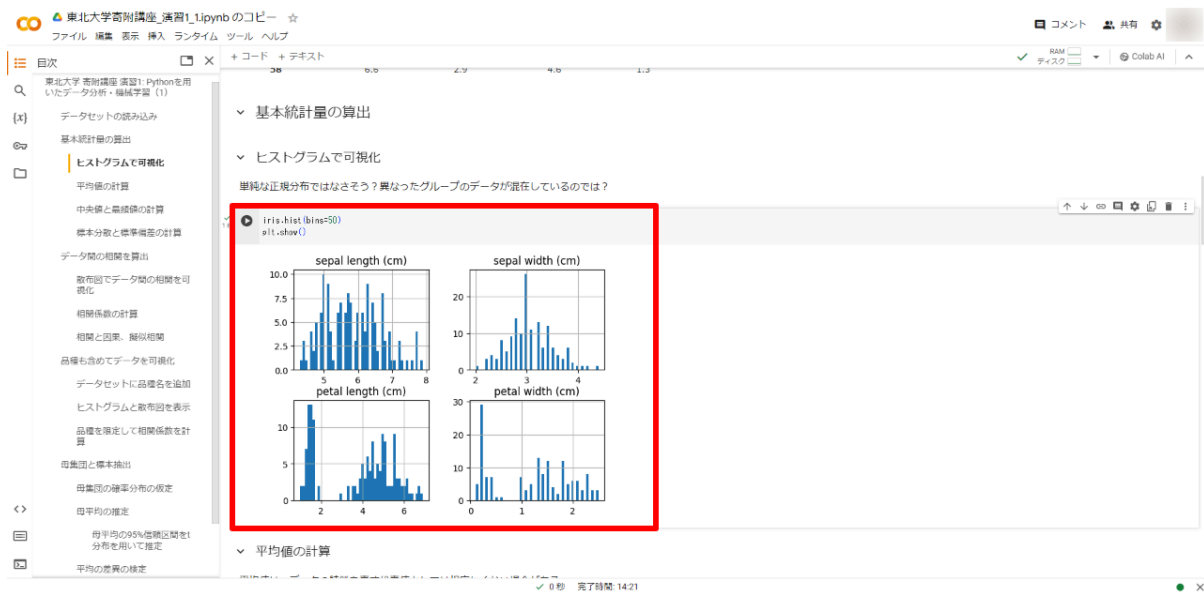
ds = datasets.load_iris()
iris = pd.DataFrame(ds.data, columns=ds.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print(f"データセットのサンプル:")
iris.sample(10)
```

The output of the code is displayed below the cell, showing the number of rows (150) and a sample of 10 rows of the Iris dataset. The output is highlighted with a red box. The table shows the following data:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
57	4.9	2.4	3.3	1.0
54	6.5	2.8	4.6	1.5
33	5.5	4.2	1.4	0.2
126	6.2	2.8	4.8	1.8
35	5.0	3.2	1.2	0.2
19	5.1	3.8	1.5	0.3
110	6.5	3.2	5.1	2.0
146	6.3	2.5	5.0	1.9
135	7.7	3.0	6.1	2.3
58	6.6	2.9	4.6	1.3

4. 演習1-1を進める

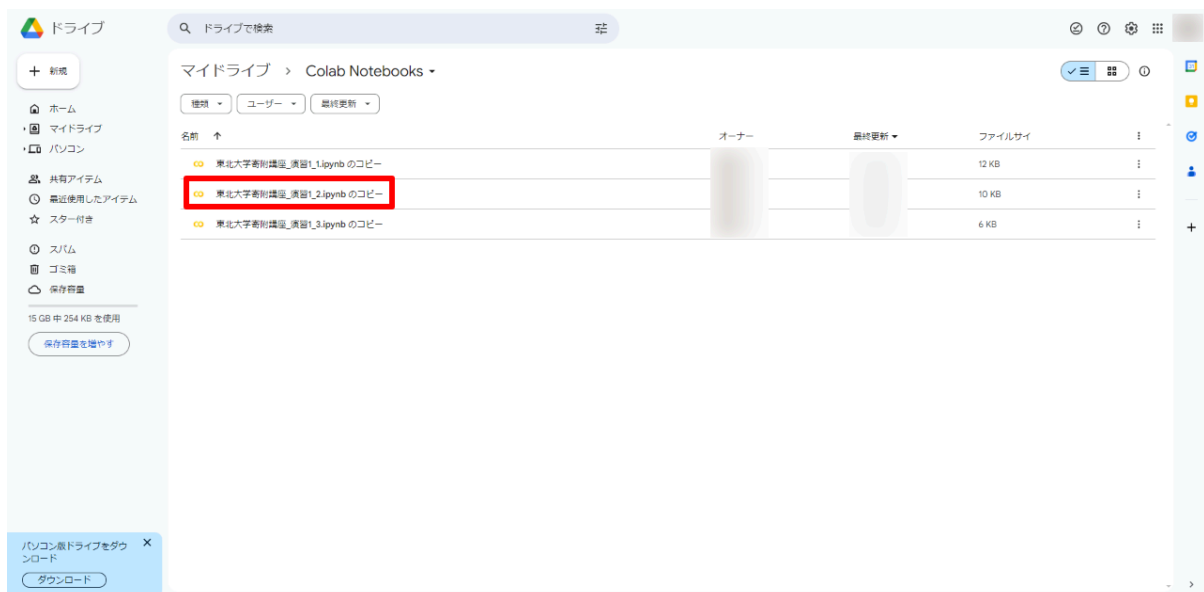
手順 2、3 と同様の手順にて、プログラム実行までの説明を呼んだ上で、プログラムの動作確認を行う



演習1-2

1. 東北大学寄附講座_演習1_2.ipynb のコピーを開く

※東北大学寄附講座_演習1_2.ipynbが開かれていない場合はGoogleドライブのマイドライブ内、「Colab Notebooks」フォルダから開く

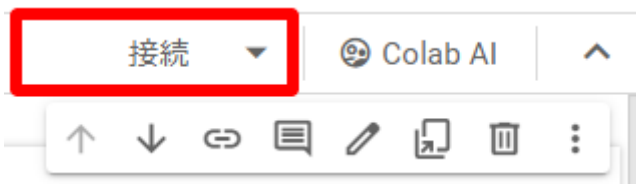


2. プログラム実行

プログラムが記載されているセルを確認し、左側のプログラム実行ボタンをクリック



※初回実行時は少し時間がかかるので、画面右上のステータスが「RAMディスク」になるまで待機



1. 「接続」→ プログラム実行前
2. 「接続中」→ プログラム実行ボタンをクリック
3. 「接続済」→ プログラム実行準備完了
4. 「RAMディスク」→ プログラム実行開始

3. 実行結果の確認

プログラムが記載されたセルの下に、実行結果が表示されるので確認



4. 演習1-2を進める

手順 2、3 と同様の手順にて、プログラム実行までの説明を呼んだ上で、プログラムの動作確認を行う



補足

「SVMを用いた分類器の学習」セクションにて正答率は若干低下しているとの記載があるがテスト用データのパターンによっては正答率が高くなる場合もある

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
pca = PCA()

X_pca = pca.fit_transform(scaler.fit_transform(ds.data))
pca iris = pd.DataFrame(X_pca, columns=['第1主成分(PC1)', '第2主成分(PC2)', '第3主成分(PC3)', '第4主成分(PC4)'])
cr1, cr2, cr3, cr4 = pca.explained_variance_ratio_
cr1, cr2, cr3, cr4 = pca.explained_variance_ratio_.cumsum()

print(f"第1主成分の寄与率:{cr1:.3f}, 第2主成分の寄与率:{cr2:.3f} 第3主成分までの累積寄与率:{cr3:.3f}")

# SVMを用いた分類器の学習
# 次元削減をせずにでも情報量を失ったことで、正答率は若干低下している

data_train, data_test, label_train, label_test = model_selection.train_test_split(pca_iris.iloc[:, :2], pd.Series(data_test.target))

print(f"学習用データ数: {len(data_train)}, テスト用データ数: {len(data_test)}")
print(f"学習用データとそのラベル\n{data_train.head()}")
print(f"テスト用データとそのラベル\n{data_test.head()}")

classifier = svm.SVC()
classifier.fit(data_train, label_train)
predicted = classifier.predict(data_test)
accuracy = metrics.accuracy_score(label_test, predicted)
print(f"分類器の正答率: {accuracy:.3f}")

# 分類器の決定境界の可視化
# SVMで学習した決定境界は、実際の品種を概ねうまく区分けできていることがわかる

import numpy as np

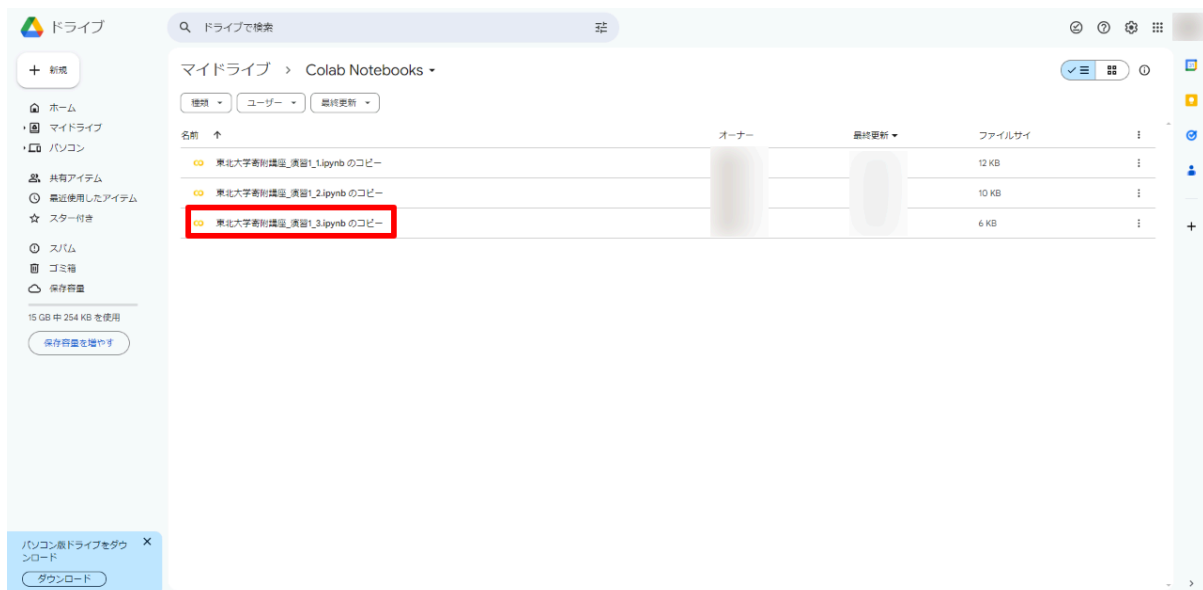
actual_pca_iris = pca_iris.iloc[:, :2].copy()
actual_pca_iris['actual'] = pd.Series(data_test.target)

fig, ax = plt.subplots(figsize=(6, 4))
ax.set_title("Decision boundary")
ax.set_xlabel("PC1")
ax.set_ylabel("PC2")
```

演習1-3

1. 東北大学寄附講座_演習1_3.ipynb のコピー を開く

※東北大学寄附講座_演習1_3.ipynbが開かれていない場合はGoogleドライブのマイドライブ内、「Colab Notebooks」フォルダから開く

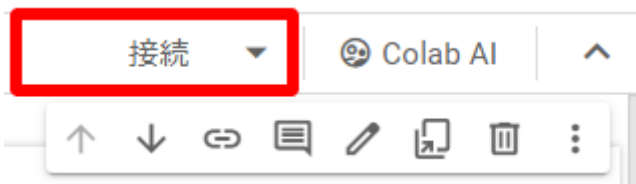


2. プログラム実行

プログラムが記載されているセルを確認し、左側のプログラム実行ボタンをクリック

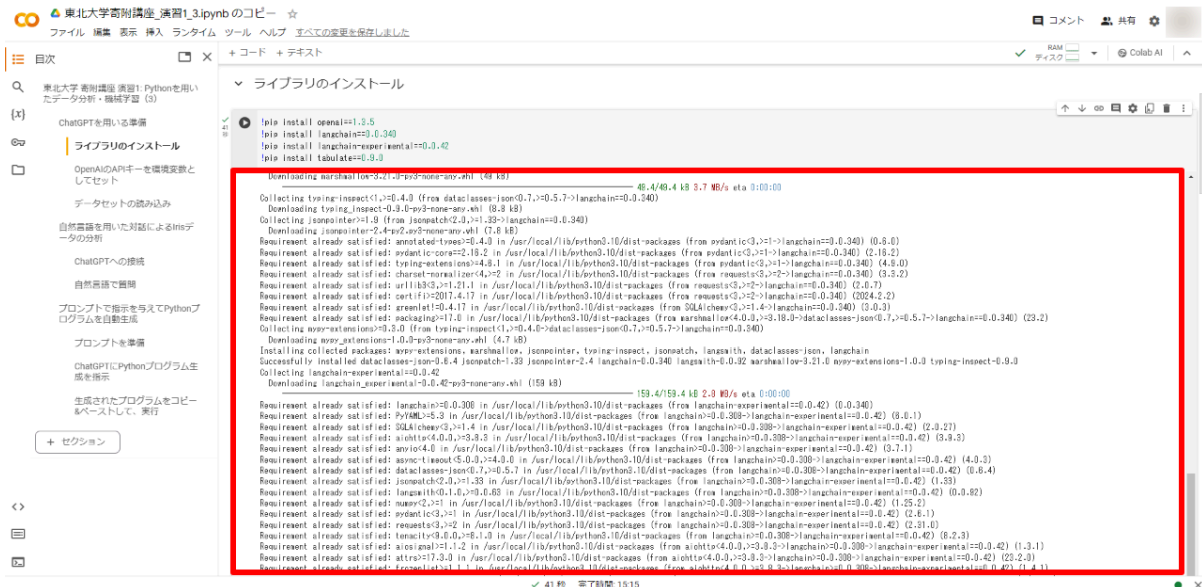


※初回実行時は少し時間がかかるので、画面右上のステータスが「RAMディスク」になるまで待機



1. 「接続」→ プログラム実行前
2. 「接続中」→ プログラム実行ボタンをクリック
3. 「接続済」→ プログラム実行準備完了
4. 「RAMディスク」→ プログラム実行開始

プログラムが記載されたセルの下に、実行結果が表示されるので確認

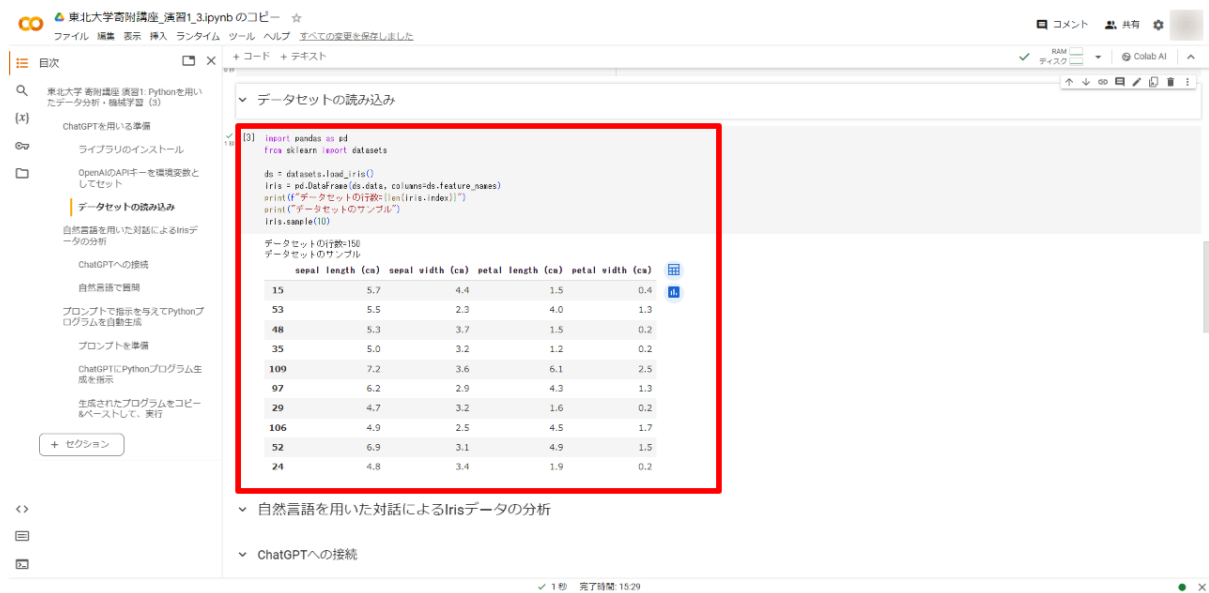


「OpenAIのAPIキーを環境変数としてセット」のセクションで個人のAPIキーを設定する



5. 演習1-3を進める

手順 2、3 と同様の手順にて、プログラム実行までの説明を呼んだ上で、プログラムの動作確認を行う



The screenshot shows a Jupyter Notebook titled "東北大学寄附講座_演習1_3.ipynb のコピー". The left sidebar contains a table of contents with items like "データセットの読み込み" and "自然言語を用いた対話によるIrisデータの分析". The main area displays a code cell with the following Python code:

```
[3]: import pandas as pd
from sklearn import datasets

ds = datasets.load_iris()
iris = pd.DataFrame(ds.data, columns=ds.feature_names)
print(f"データセットの行数: {len(iris.index)}")
print(f"データセットのサンプル:")
iris.sample(10)
```

Below the code, the output shows the dataset's dimensions and a preview of the data:

データセットの行数: 150
データセットのサンプル

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
15	5.7	4.4	1.5	0.4
53	5.5	2.3	4.0	1.3
48	5.3	3.7	1.5	0.2
35	5.0	3.2	1.2	0.2
109	7.2	3.6	6.1	2.5
97	6.2	2.9	4.3	1.3
29	4.7	3.2	1.6	0.2
106	4.9	2.5	4.5	1.7
52	6.9	3.1	4.9	1.5
24	4.8	3.4	1.9	0.2

The bottom of the interface shows a status bar with "1秒 完了時間: 15:29".