探究学習のための
機械学習 × GIS ハンズオン

新井 崇弘[1], 廣澤 聖士[2]

1. 多摩大学 経営情報学部　専任講師
2. 桐蔭横浜大学 スポーツ科学部　特任講師

2025年03月08日(土曜日)

理数系教員統計・データサイエンス授業力向上研修会

本日のハンズオンの流れ

（前半） ChatGPTを用いたクラスタリング実装

1. クラスタリングの説明
2. ChatGPTを用いた分析実装の説明（R、Pythonとの連携について）
3. Rの説明
4. Pythonの説明
5. ChatGPT上で行うクラスタリングの分析実装

（後半）ArcGIS Pro で空間解析

機械学習手法

	分析手法	主な用途
教師あり学習	回帰	予測、要因分解など
	分類	カテゴリ分類、スパム検出、画像分類など
教師なし学習	次元削減	指標構築、データの構造要約など
	クラスタリング	セグメンテーション、異常検知など

教師あり学習：入力データ（説明変数）とそれに対応する正解ラベル（目的変数）を用いてモデルを訓練し、入力データから正しい出力を予測する

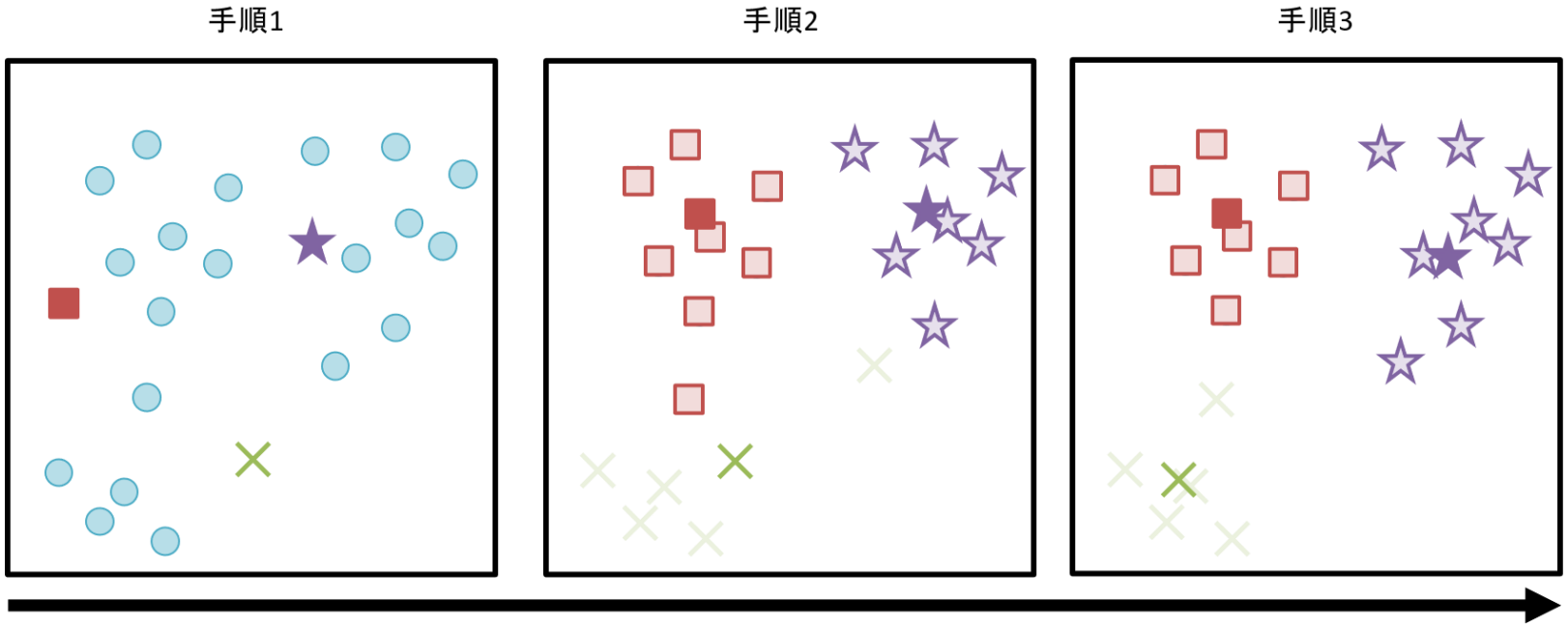
教師なし学習：ラベルのないデータから構造やパターンを見つけ出す
➡今回は、クラスタリング（k-means法）について実装を行う

k-means法

クラスタリング:データの特徴をもとに、似た特性を持つデータ同士をまとめて「クラスタ」と呼ばれるグループに分け、各グループの共通特徴を明らかにする手法。

- 階層的クラスタリング
- k-means法

などがある



1) あらかじめクラスタ数kを決定し、ランダムにk個のセントロイド(代表点)を選択する
2) 各データポイントを最も近いセントロイドに割り当てる
3) 各クラスタの平均値を新しいセントロイドとして計算する
4) セントロイドの位置が変わらなくなるまで2)および3)の手順を繰り返す

代表点や距離の決定

【クラスタを統合する際の代表点を定める方法】

- 重心法： クラスタ内のすべてのデータの各次元の平均値を計算し、その平均値を代表点とする
- メドイド法： クラスタ内の各データ点との距離の総和が最小となる実際のデータ点を代表点とする
- ウォード法: クラスタ内の分散を最小化するように統合する

など

【データ間の距離】

- ユークリッド距離： 空間上の2点間の直線距離
- マンハッタン距離： 直交する方向（縦横）の移動距離の合計
- コサイン類似度： 2つのベクトル間のなす角の余弦を用いて、方向性（向き）の類似性を計算

など

ChatGPTの出力では、特に指定しなければ、重心法/ウォード法およびユークリッド距離が使用される場合が多い

クラスタ数の選択

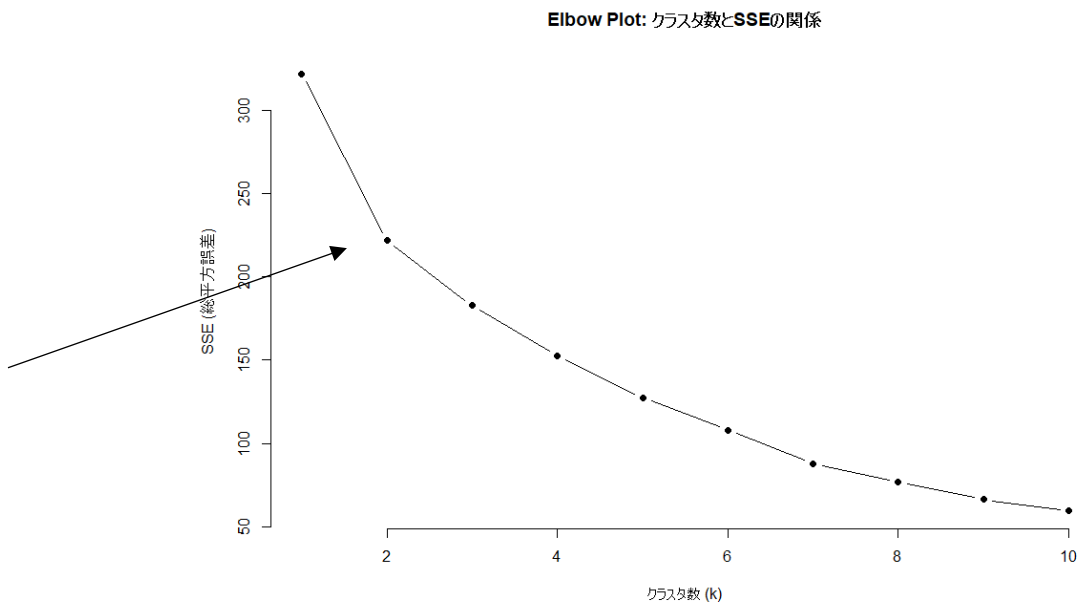
- エルボー法

クラスタ数kを変化させながら、クラスタ内誤差の平方和(Sum of Squared Errors: SSE)を計算し、その変化を可視化する手法。

クラスタ数kが増加するにつれてSSEは減少し、その減少率が急激に緩やかになる「肘」の位置を適切なクラスタ数とする。

(例)

このような場合にはクラスタ2を検討する



<u>最終的には分析者自身が「どのクラスタ分けが有益か」といった視点（リサーチの目的）に合わせて決定する</u>

ChatGPTを用いた実装の説明

今回使用するデータセット　climate_data.csv

- 各市内の地上気象観測を行っている気象台等のデータを収録（計47都道府県）
- 1991～2020年の30年間の平均値（気温、気圧、湿度、風速、日照時間、降水量、積雪量）

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	地域コード	都道府県	市	平均気温	平均現地気圧	平均相対湿度	平均風速	日照時間の合計	降水量の合計	降雪量の合計	緯度	経度	標高
2	R01100	北海道	札幌市	9.2	1009.2	69	3.6	1718	1146.1	479	43.06	141.33	17.4
3	R02201	青森県	青森市	10.7	1012.9	75	3.7	1589.2	1350.7	567	40.82	140.77	2.8
4	R03201	岩手県	盛岡市	10.6	995.3	74	2.9	1686.3	1279.9	209	39.7	141.17	155.2
5	R04100	宮城県	仙台市	12.8	1008.8	71	3.2	1836.9	1276.7	59	38.26	140.9	38.9
6	R05201	秋田県	秋田市	12.1	1011.3	73	4.3	1527.4	1741.6	273	39.72	140.1	6.3
7	R06201	山形県	山形市	12.1	995.9	74	1.7	1617.9	1206.7	285	38.26	140.35	152.5
8	R07201	福島県	福島市	13.4	1005.9	69	2.4	1753.8	1207	122	37.76	140.47	67.4
9	R08201	茨城県	水戸市	14.1	1010.3	74	2.3	2000.8	1367.7	12	36.38	140.47	29
10	R09201	栃木県	宇都宮市	14.3	997.2	70	2.9	1961.1	1524.7	18	36.55	139.87	119.4
11	R10201	群馬県	前橋市	15	1000.4	62	2.4	2153.7	1247.4	19	36.41	139.06	112.1
12	R11202	埼玉県	熊谷市	15.4	1010.1	65	2.5	2106.6	1305.8	16	36.15	139.38	30
13	R12100	千葉県	千葉市	16.2	1013.1	68	3.9	1945.5	1454.7	7	35.6	140.1	3.5
14	R13100	東京都	東京都区部	15.8	1010.9	65	2.9	1926.7	1598.2	8	35.69	139.75	25.2
15	R14100	神奈川県	横浜市	16.2	1008.6	67	3.5	2018.3	1730.8	9	35.44	139.65	39.1
16	R15100	新潟県	新潟市	13.9	1013.7	72	3.3	1639.6	1845.9	139	37.89	139.02	4.1
17	R16201	富山県	富山市	14.5	1012.9	76	2.9	1647.2	2374.2	253	36.71	137.2	8.6
18	R17201	石川県	金沢市	15	1010.8	70	4	1714.1	2401.5	157	36.59	136.63	5.7
19	R18201	福井県	福井市	14.8	1013	75	2.8	1653.7	2299.6	186	36.06	136.22	8.8
20	R19201	山梨県	甲府市	15.1	980.6	64	2.2	2225.8	1160.7	23	35.67	138.55	272.8
21	R20201	長野県	長野市	12.3	965.2	72	2.5	1969.9	965.1	163	36.66	138.19	418.2
22	R21201	岐阜県	岐阜市	16.2	1012.6	66	2.6	2108.6	1860.7	34	35.4	136.76	12.7
23	R22100	静岡県	静岡市	16.9	1011.6	68	2.2	2151.5	2327.3	0	34.98	138.4	14.1

本データは、SSDSE（教育用標準データセット）のうち 6.SSDSE-気候値（SSDSE-F）から一部抜粋したものです。
出所：https://www.nstac.go.jp/use/literacy/ssdse/

ChatGPT上で実装と留意点

ChatGPT上で実装することが可能だが、正確性や再現性の観点から、念のためRやPythonなどのソースコードを確認することが重要。

以下のURLからダウンロード可能

https://cran.r-project.org/bin/windows/base/

R-4.4.3 for Windows

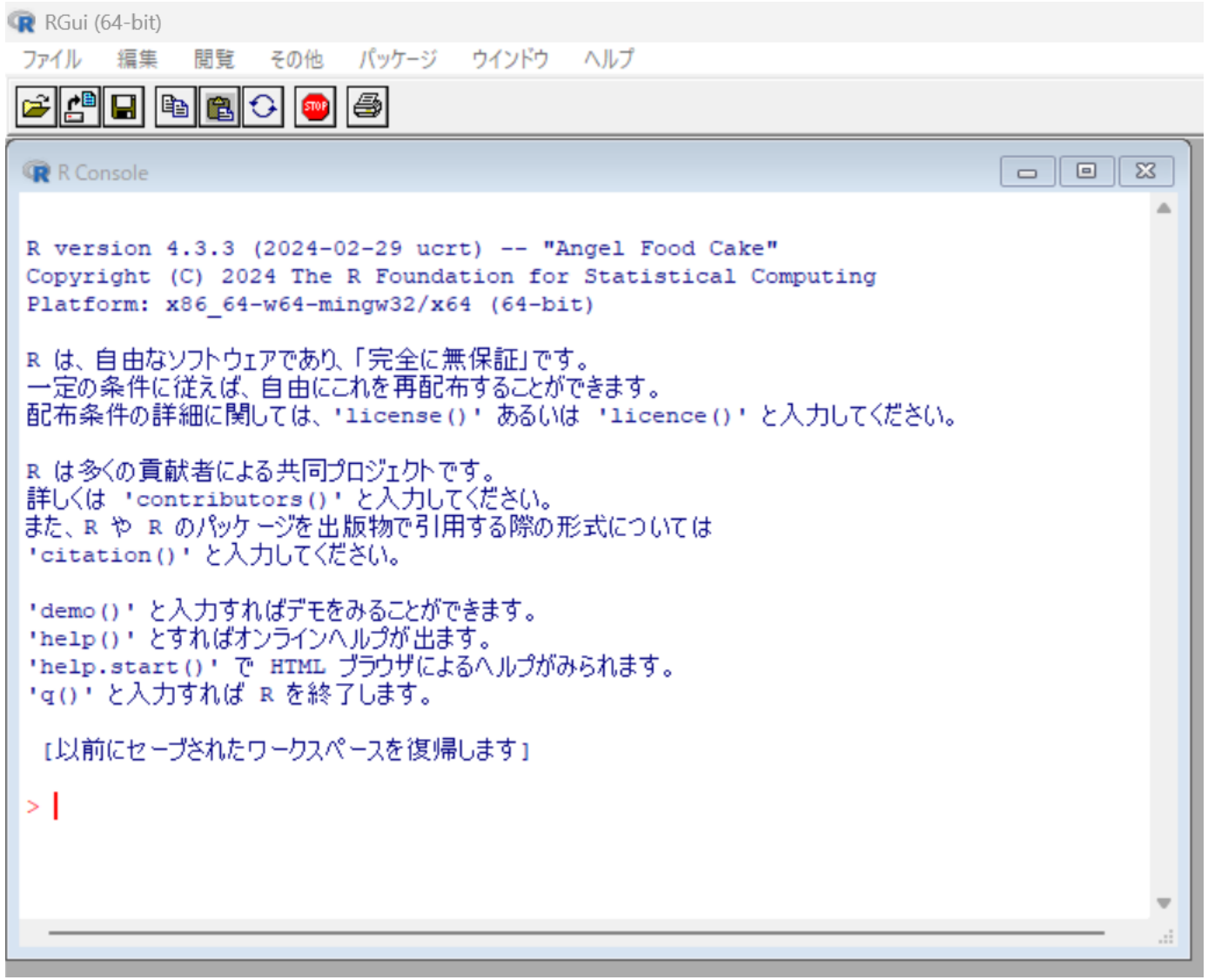
Download R-4.4.3 for Windows (85 megabytes, 64 bit)

README on the Windows binary distribution
New features in this version

This build requires UCRT, which is part of Windows since Windows 10 and Windows Server 2016. On older systems, UCRT has to be installed manually from here.

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the md5sum of the .exe to the fingerprint on the master server.

ChatGPT上で実装と留意点



Rをインストールすると左のようなコンソール画面が表示される

この画面に、コードを入力して分析を実装することができる

ChatGPT上で実装と留意点

データを読み込ませる際には、

- パスを使ったファイルの指定

climate_data <- read.csv(“C:/Users/……/climate_data.csv”)

文字化けする場合には下記を指定

climate_data <- read.csv(“C:/Users/……/climate_data.csv”, fileEncoding = ”CP932”)

k-meansクラスタリングのプロンプト

分析に適した好みのLLMを選んでください
k-meansクラスタリングではそこまで大きな差は出ません

※無料版の場合には、利用回数によって制限がかかる場合がありますのでご注意ください

k-meansクラスタリングのプロンプトとRソースコードの出力

以下は、クラスタ結果を含むデータセットの出力例とCSV保存を含めたRのソースコードになります。

r コピーする

```r
# CSVファイルの読み込み
climate_data <- read.csv("climate_data.csv")

# ここでは「平均気温」「平均現地気圧」「平均相対湿度」「平均風速」「日照時間の合計」
vars <- c("平均気温", "平均現地気圧", "平均相対湿度",
          "平均風速", "日照時間の合計", "降水量の合計", "降雪量の合計")
data_scaled <- scale(climate_data[, vars])

# --- k-meansクラスタリングの実施 ---
set.seed(123)  # 再現性のため

# クラスタ数2の場合
km2 <- kmeans(data_scaled, centers = 2, nstart = 25)
# クラスタ数3の場合
km3 <- kmeans(data_scaled, centers = 3, nstart = 25)
# クラスタ数4の場合
km4 <- kmeans(data_scaled, centers = 4, nstart = 25)
# クラスタ数5の場合
km5 <- kmeans(data_scaled, centers = 5, nstart = 25)
```

解説

- **データ読み込み**: `read.csv()` 関数を用いてCSVファイルを読み込みます。
- **変数選択とスケーリング**: 指定の7変数のみを抽出し、`scale()` 関数で標準化しています。
- **k-means実施**: `kmeans()` 関数を用い、`nstart = 25` で初期値の異なる試行を複数回実施し安定した結果を得ています。

プロンプト

- 平均気温
- 平均現地気圧
- 平均相対湿度
- 平均風速
- 日照時間の合計
- 降水量の合計
- 降雪量の合計

を用いて、k-meansを実施してください。
クラスタ数は2つの場合、3つの場合、4つの場合、5つの場合で行って、データセットにクラスタIDを結合してください。そのためのソースコード（R/Python）を提示してください。
###

<u>※必ずしも同じ出力が出るとは限らないので注意！</u>

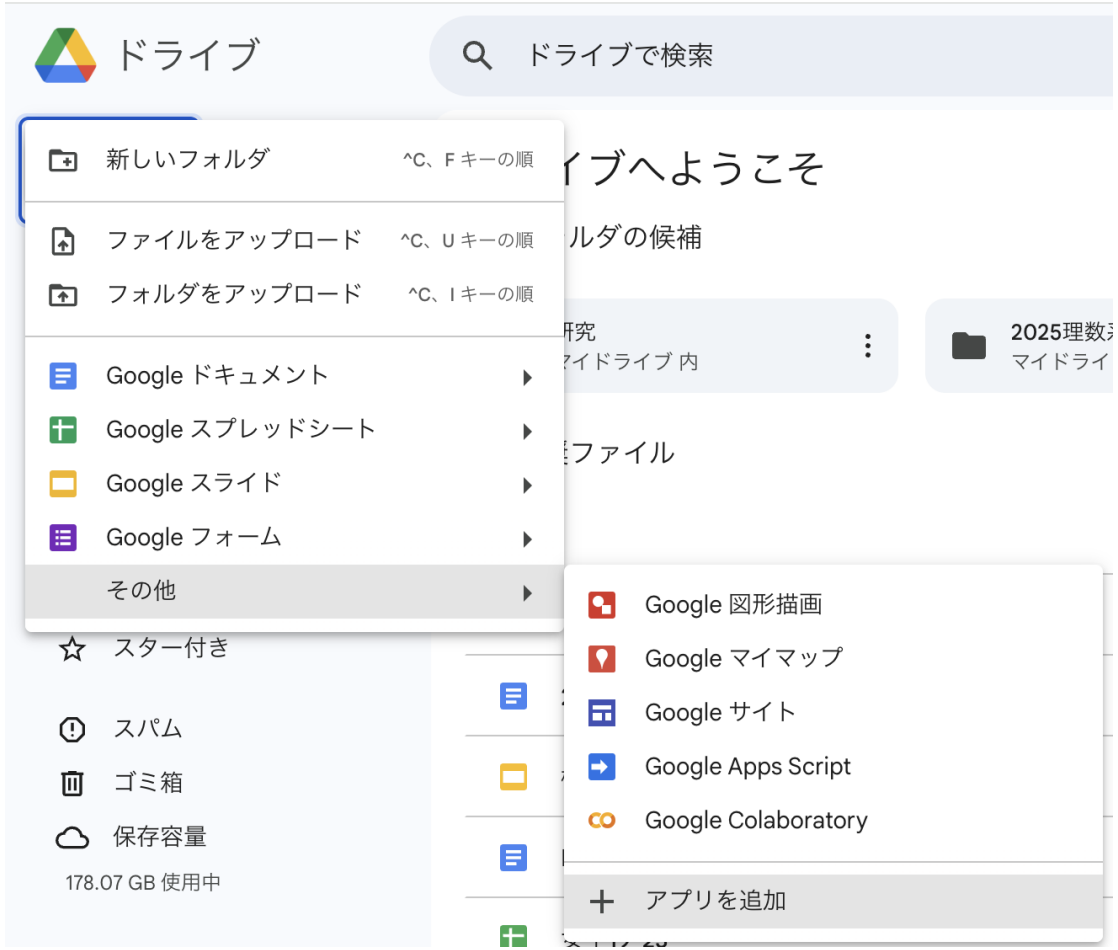
```r
# --- クラスタIDを元のデータセットに結合 ---
climate_data$cluster2 <- km2$cluster
climate_data$cluster3 <- km3$cluster
climate_data$cluster4 <- km4$cluster
climate_data$cluster5 <- km5$cluster

# --- 結果確認 ---
head(climate_data)

# --- クラスタ結果を含むデータセットをcsvとして保存 ---
write.csv(climate_data, "climate_data_with_clusters.csv", row.names = FALSE)
```

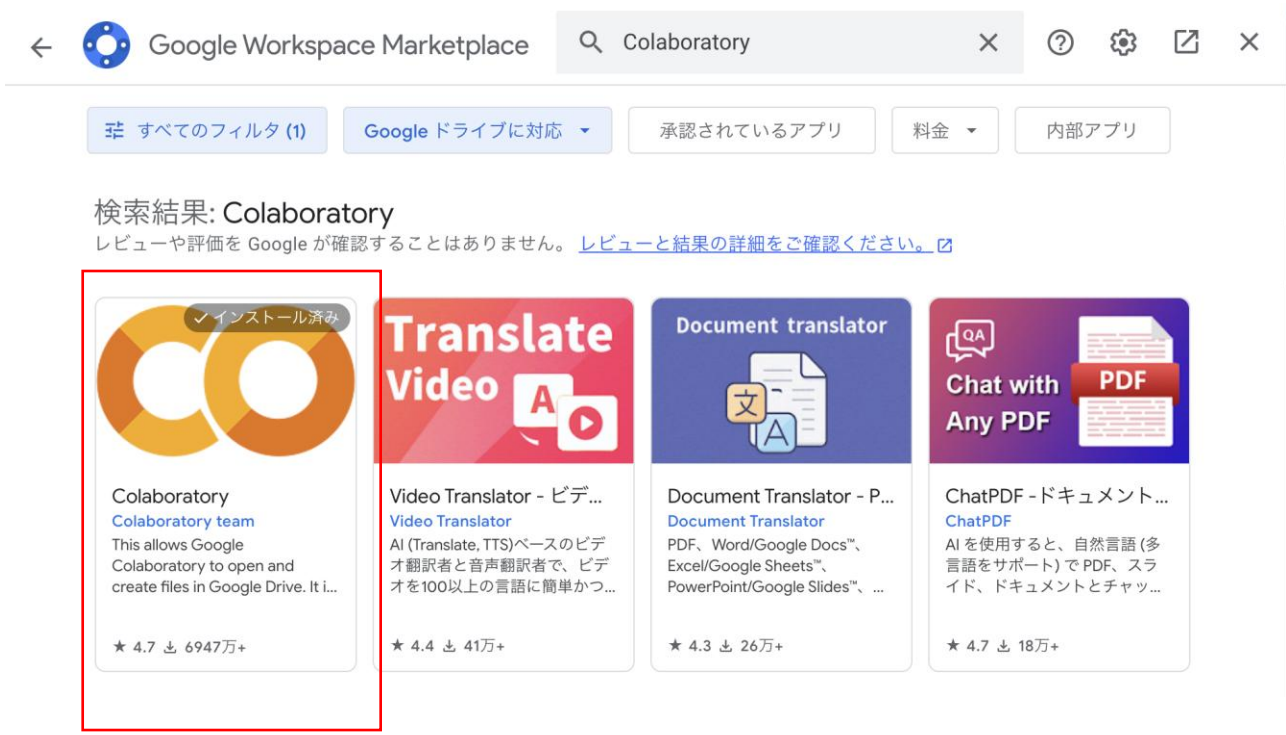
Rソースコードの確認

```
> # ここでは「平均気温」「平均現地気圧」「平均相対湿度」「平均風速」「日照時間の合計」「降水量の合計」「降雪量の合計」の
> vars <- c("平均気温", "平均現地気圧", "平均相対湿度",
+           "平均風速", "日照時間の合計", "降水量の合計", "降雪量の合計")
> data_scaled <- scale(climate_data[, vars])
>
>
> # --- k-meansクラスタリングの実施 ---
> set.seed(123)  # 再現性のため
>
> # クラスタ数2の場合
> km2 <- kmeans(data_scaled, centers = 2, nstart = 25)
> # クラスタ数3の場合
> km3 <- kmeans(data_scaled, centers = 3, nstart = 25)
> # クラスタ数4の場合
> km4 <- kmeans(data_scaled, centers = 4, nstart = 25)
> # クラスタ数5の場合
> km5 <- kmeans(data_scaled, centers = 5, nstart = 25)
>
>
> # --- クラスタIDを元のデータセットに結合 ---
> climate_data$cluster2 <- km2$cluster
> climate_data$cluster3 <- km3$cluster
> climate_data$cluster4 <- km4$cluster
> climate_data$cluster5 <- km5$cluster
>
> # --- 結果確認 ---
> head(climate_data)
  地域コード 都道府県     市 平均気温 平均現地気圧 平均相対湿度 平均風速 日照時間の合計 降水量の合計 降雪量の合計
1     R01100   北海道 札幌市      9.2       1009.2           69      3.6         1718.0       1146.1          479
2     R02201   青森県 青森市     10.7       1012.9           75      3.7         1589.2       1350.7          567
3     R03201   岩手県 盛岡市     10.6        995.3           74      2.9         1686.3       1279.9          209
4     R04100   宮城県 仙台市     12.8       1008.8           71      3.2         1836.9       1276.7           59
5     R05201   秋田県 秋田市     12.1       1011.3           73      4.3         1527.4       1741.6          273
6     R06201   山形県 山形市     12.1        995.9           74      1.7         1617.9       1206.7          285
   緯度    経度  標高 cluster2 cluster3 cluster4 cluster5
1 43.06 141.33  17.4        1        1        3        3
2 40.82 140.77   2.8        1        1        3        3
3 39.70 141.17 155.2        1        1        3        3
4 38.26 140.90  38.9        1        1        2        4
5 39.72 140.10   6.3        1        1        3        3
6 38.26 140.35 152.5        1        1        3        3
```

```
# CSVファイルの読み込み
climate_data <- read.csv("C:/Users/……/climate_data.csv", fileEncoding = "CP932")

# ここでは「平均気温」「平均現地気圧」「平均相対湿度」「平均風速」「日照時間の合計」「降水量の合計」「降雪量の合計」の7変数だけを使用します
vars <- c("平均気温", "平均現地気圧", "平均相対湿度",
          "平均風速", "日照時間の合計", "降水量の合計", "降雪量の合計")
data_scaled <- scale(climate_data[, vars])

# --- k-meansクラスタリングの実施 ---
set.seed(123) # 再現性のため

# クラスタ数2の場合
km2 <- kmeans(data_scaled, centers = 2, nstart = 25)
# クラスタ数3の場合
km3 <- kmeans(data_scaled, centers = 3, nstart = 25)
# クラスタ数4の場合
km4 <- kmeans(data_scaled, centers = 4, nstart = 25)
# クラスタ数5の場合
km5 <- kmeans(data_scaled, centers = 5, nstart = 25)

# --- クラスタIDを元のデータセットに結合 ---
climate_data$cluster2 <- km2$cluster
climate_data$cluster3 <- km3$cluster
climate_data$cluster4 <- km4$cluster
climate_data$cluster5 <- km5$cluster

# --- 結果確認 ---
head(climate_data)

# --- クラスタ結果を含むデータセットをCSVとして保存 ---
write.csv(climate_data, ("C:/Users/……/climate_data.csv", fileEncoding = "cp932")
```

Pythonの場合：Google colaboratoryを使って環境構築ができる





ご自身のGoogle Driveにアクセスし、「＋新規」→「その他」→「アプリを追加」→
「Colaboratory」をインストールすると画面のようにファイルが作成できるようになります

Python出力コードミニ解説

For文：繰り返し処理を使う

```
[]  # クラスタリングを実施（クラスタ数 2, 3, 4, 5）
    # クラスタが2~6で同じ処理を行うため、別々にコードを書かずfor文を用いて繰り返し処理を行っている。

    # 文字kが2~6それぞれに変化するイメージ
    for k in range(2, 6):
        # 最初の処理ではk=2が入る、二回目の処理ではk=3となり、6まで繰り返すイメージ
        kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
        # 標準化後のデータ(X_scaled)に対してkmeansを実行。その結果を元のdfに結合している。
        df[f"Cluster_{k}"] = kmeans.fit_predict(X_scaled)
```

For文なし（クラスター数2の場合）

```
# for文で書かずn=2でやってみる
# 最初の処理では今回のようにk=2が入る、二回目の処理ではk=3となり、6まで繰り返すイメージ
kmeans = KMeans(n_clusters=2, random_state=42, n_init=10)
# fをつけることでkが繰り返されるようになっていたが、今回は文字列をそのまま指定している
df["Cluster_2"] = kmeans.fit_predict(X_scaled)
```

列の結合

```
# pythonは以下のようにすることで新しい列を追加することができる
# データを読み込んだときにdf=にしているため冒頭はdfになる
df[新しい列名]=データ
```

```
1   import pandas as pd
2   from sklearn.cluster import KMeans
3   from sklearn.preprocessing import StandardScaler
4
5   # CSVファイルの読み込み（エンコーディングを適宜変更）
6   file_path = "climate_data.csv"
7   df = pd.read_csv(file_path, encoding="shift_jis")
8
9   # クラスタリングに使用する列を選択
10  features = [
11      "平均気温", "平均現地気圧", "平均相対湿度", "平均風速",
12      "日照時間の合計", "降水量の合計", "降雪量の合計"
13  ]
14  X = df[features]
15
16  # 特徴量の標準化（スケーリング）
17  scaler = StandardScaler()
18  X_scaled = scaler.fit_transform(X)
19
20  # クラスタリングを実施（クラスタ数 2, 3, 4, 5）
21  for k in range(2, 6):
22      kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
23      df[f"Cluster_{k}"] = kmeans.fit_predict(X_scaled)
24
25  # クラスタリング結果をCSVに保存
26  df.to_csv("clustered_climate_data.csv", index=False, encoding="shift_jis")
27
28  # クラスタリング結果の表示
29  print(df.head())
30
```

ChatGPT上で実行する際のプロンプト

※今回はChatGPT 4o を使用しています

ChatGPT 4o

1	1	R01100	北海道	札幌市
2	2	R02201	青森県	青森市
3	3	R03201	岩手県	盛岡市
4	4	R04100	宮城県	仙台市
5	5	R05201	秋田県	秋田市
6	6	R06201	山形県	山形市

平均気温
平均現地気圧
平均相対湿度
平均風速
日照時間の合計
降水量の合計
降雪量の合計
を用いて、k-meansを実施してください。
クラスタ数は２つの場合、３つの場合、４つの場合、５つの場合で行って、
データセットにクラスタIDを結合してください。
ｃｓｖ形式でダウンロードできるようにしてください。
これらをChatGPT上で実行してください。

＃＃＃　プロンプト
- **平均気温**
- **平均現地気圧**
- **平均相対湿度**
- **平均風速**
- **日照時間の合計**
- **降水量の合計**
- **降雪量の合計**

を用いて、k-meansを実施してください。
クラスタ数は2つの場合、3つの場合、4つの場合、5つの場合で行って、データセットにクラスタIDを結合してください。
csv形式でダウンロードできるようにしてください。
これらをChatGPT上で実行してください。
＃＃＃

※ ChatGPT上で実行する際、文字化けする場合には、climate data UTF.csv　を使用してみてください

もしRやPythonなどに馴染みのない方は、ぜひChatGPT上で分析実装してみてください。

ChatGPT上での実行結果

出力結果として、分析済みデータのダウンロードリンクを提示するだけの場合もある。

分析プロセスを表示するにはここをクリックするとPythonのコードが表示される。
ソースコードをチェックしながら分析を進める必要がある。

もしダウンロード結果が文字化けしてしまう場合には、
(例)エンコーディング(文字コード)をCP932としてダウンロードしてください
などと指示すると良いでしょう。

K-meansクラスタリングの出力結果

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Unnamed	地域コード	都道府県	市	平均気温	平均現地気	平均相対湿	平均風速	日照時間の	降水量の合	降雪量の合	緯度	経度	標高	Cluster_2	Cluster_3	Cluster_4	Cluster_5
2	1	R01100	北海道	札幌市	9.2	1009.2	69	3.6	1718	1146.1	479	43.06	141.33	17.4	0	2	0	2
3	2	R02201	青森県	青森市	10.7	1012.9	75	3.7	1589.2	1350.7	567	40.82	140.77	2.8	0	2	0	2
4	3	R03201	岩手県	盛岡市	10.6	995.3	74	2.9	1686.3	1279.9	209	39.7	141.17	155.2	0	2	0	2
5	4	R04100	宮城県	仙台市	12.8	1008.8	71	3.2	1836.9	1276.7	59	38.26	140.9	38.9	0	2	3	3
6	5	R05201	秋田県	秋田市	12.1	1011.3	73	4.3	1527.4	1741.6	273	39.72	140.1	6.3	1	1	2	1
7	6	R06201	山形県	山形市	12.1	995.9	74	1.7	1617.9	1206.7	285	38.26	140.35	152.5	0	2	0	2
8	7	R07201	福島県	福島市	13.4	1005.9	69	2.4	1753.8	1207	122	37.76	140.47	67.4	0	2	0	2
9	8	R08201	茨城県	水戸市	14.1	1010.3	74	2.3	2000.8	1367.7	12	36.38	140.47	29	0	2	3	3
10	9	R09201	栃木県	宇都宮市	14.3	997.2	70	2.9	1961.1	1524.7	18	36.55	139.87	119.4	0	1	3	4
11	10	R10201	群馬県	前橋市	15	1000.4	62	2.4	2153.7	1247.4	19	36.41	139.06	112.1	0	2	3	3
12	11	R11202	埼玉県	熊谷市	15.4	1010.1	65	2.5	2106.6	1305.8	16	36.15	139.38	30	0	2	3	3
13	12	R12100	千葉県	千葉市	16.2	1013.1	68	3.9	1945.5	1454.7	7	35.6	140.1	3.5	0	2	3	4
14	13	R13100	東京都	東京都区部	15.8	1010.9	65	2.9	1926.7	1598.2	8	35.69	139.75	25.2	0	1	2	4
15	14	R14100	神奈川県	横浜市	16.2	1008.6	67	3.5	2018.3	1730.8	9	35.44	139.65	39.1	0	1	2	4
16	15	R15100	新潟県	新潟市	13.9	1013.7	72	3.3	1639.6	1845.9	139	37.89	139.02	4.1	1	1	2	1
17	16	R16201	富山県	富山市	14.5	1012.9	76	2.9	1647.2	2374.2	253	36.71	137.2	8.6	1	0	1	1
18	17	R17201	石川県	金沢市	15	1010.8	70	4	1714.1	2401.5	157	36.59	136.63	5.7	1	0	1	0
19	18	R18201	福井県	福井市	14.8	1013	75	2.8	1653.7	2299.6	186	36.06	136.22	8.3	1	0	1	1
20	19	R19201	山梨県	甲府市	15.1	980.6	64	2.2	2225.8	1160.7	23	35.67	138.55	272.8	0	2	3	3
21	20	R20201	長野県	長野市	12.3	965.2	72	2.5	1969.9	965.1	163	36.66	138.19	418.2	0	2	0	3
22	21	R21201	岐阜県	岐阜市	16.2	1012.6	66	2.6	2108.6	1860.7	34	35.4	136.76	12.7	1	1	2	4
23	22	R22100	静岡県	静岡市	16.9	1011.6	68	2.2	2151.5	2327.3	0	34.98	138.4	14.1	1	0	1	0

クラスタIDが元データに結合されていれば成功です

※クラスタIDは必ずしも皆同じ結果になるとは限らないので注意してください

後半はこのデータセットを用いてArcGIS Pro で空間解析を行います