Pythonによるデータ解析

2年4組55番
三好　駿

覚えていますか？

春学期のR言語の輪講の回

そう、主成分分析

僕が担当した回です

しっかり教科書を読みこんだもののあまり理解できず、当日はほとんど先生に解説してもらい「だそうです」の連発…

そこで！

今回は主成分分析のリベンジをしたい！
ということでPythonで主成分分析をしてみました
ついでにRとPythonの主成分分析を軸にした比較もしてみました

主成分分析をしたデータは皆さんおなじみのirisです

（主成分分析とは）

・データ解析手法の一つ

・たくさんの量的な説明変数を、より少ない指標や合成変数（複数の変数が合体したもの）に要約する手法

・要する多次元の要素を低次元に圧縮

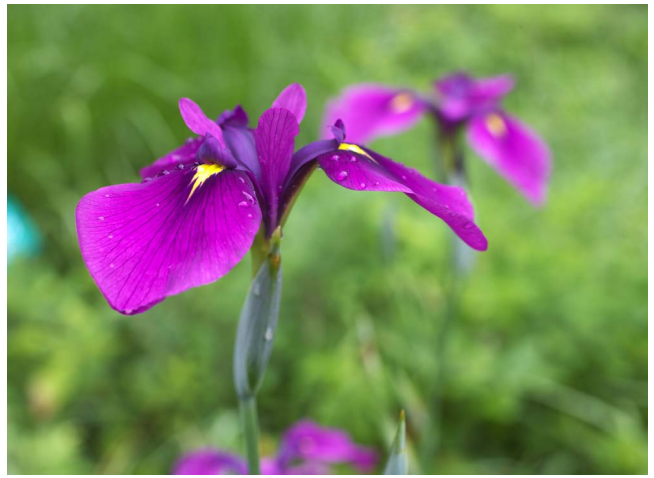
irisとは　わすれてしまった人もいると思うので

データセットiris

・irisは花のアヤメのデータを集めたもの

・setosa、versicolor、virginicaの三種類のアヤメを4つの特徴量についてそれぞれ50個ずつ集めたもの

sepal length (cm)	がく片の長さ
sepal width (cm)	がく片の幅
petal length (cm)	花弁の長さ
petal width (cm)	花弁の幅



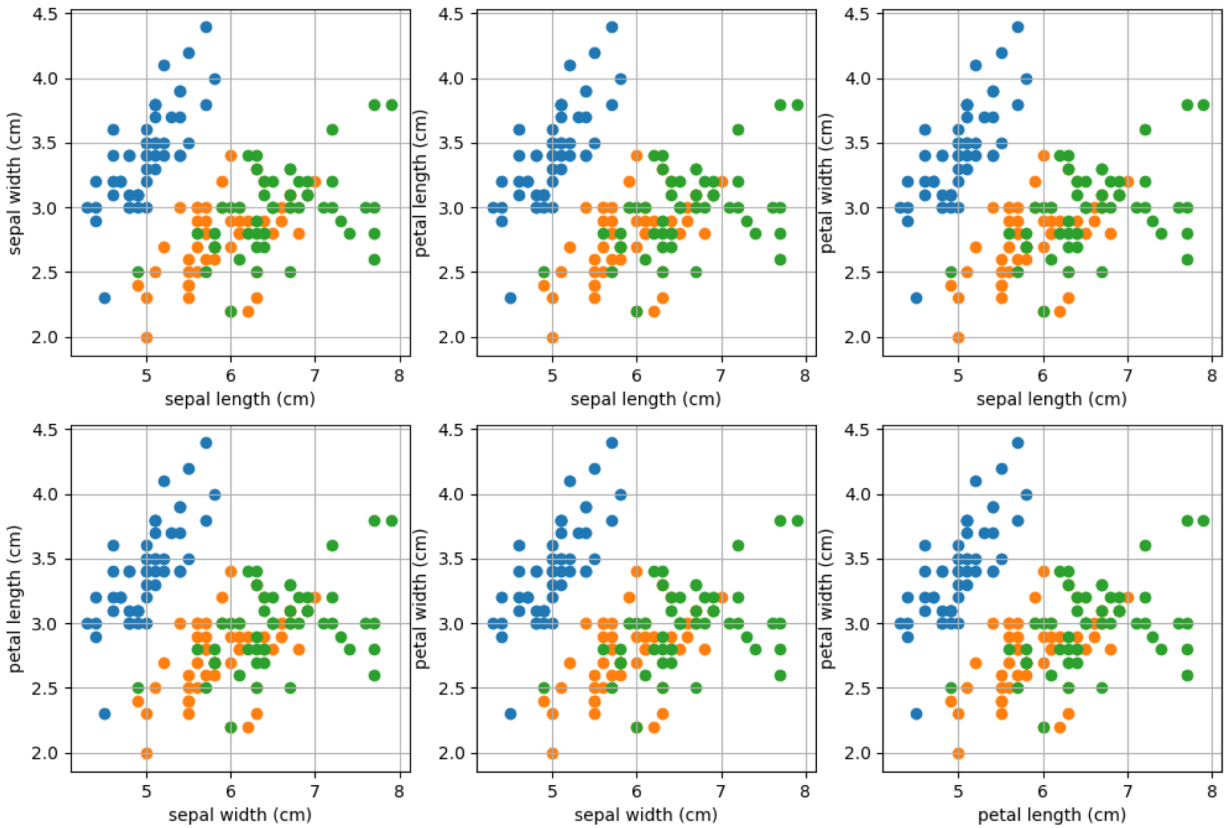
irisとは

データの中身をRで表示するとこんな感じ
とは言っても見にくいので視覚化してみると

Console ~/

```
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
16           5.7         4.4          1.5         0.4     setosa
17           5.4         3.9          1.3         0.4     setosa
18           5.1         3.5          1.4         0.3     setosa
19           5.7         3.8          1.7         0.3     setosa
20           5.1         3.8          1.5         0.3     setosa
21           5.4         3.4          1.7         0.2     setosa
22           5.1         3.7          1.5         0.4     setosa
23           4.6         3.6          1.0         0.2     setosa
24           5.1         3.3          1.7         0.5     setosa
25           4.8         3.4          1.9         0.2     setosa
26           5.0         3.0          1.6         0.2     setosa
27           5.0         3.4          1.6         0.4     setosa
28           5.2         3.5          1.5         0.2     setosa
29           5.2         3.4          1.4         0.2     setosa
30           4.7         3.2          1.6         0.2     setosa
31           4.8         3.1          1.6         0.2     setosa
32           5.4         3.4          1.5         0.4     setosa
33           5.2         4.1          1.5         0.1     setosa
34           5.5         4.2          1.4         0.2     setosa
35           4.9         3.1          1.5         0.2     setosa
36           5.0         3.2          1.2         0.2     setosa
37           5.5         3.5          1.3         0.2     setosa
38           4.9         3.6          1.4         0.1     setosa
39           4.4         3.0          1.3         0.2     setosa
40           5.1         3.4          1.5         0.2     setosa
41           5.0         3.5          1.3         0.3     setosa
42           4.5         2.3          1.3         0.3     setosa
```

irisとは



特徴量が4つあるためそれぞれを比較した6つのグラフが出てきてしまう
→主成分分析をして4次元を2次元に圧縮し、1つのグラフにまとめる！

（視覚化にはライブラリmatplotlibを使用：詳しくは後述）

主成分分析

早速このデータセットirisをPythonで主成分分析して行く

<手順>

①Pythonでの主成分分析はscikit-learnというライブラリを使って行う（irisのデータセットもscikit-learnに入っている）

②ライブラリmatplotlibを使って視覚化

主成分分析

①Pythonでの主成分分析はscikit-learというライブラリを使って行う（irisのデータセットもscikit-learnに入っている）

ライブラリーのインポート

```
from sklearn.decomposition import PCA    →主成分分析をするツール
from sklearn import datasets             →データセット
```

主成分分析

①Pythonでの主成分分析はscikit-learというライブラリを使って行う（irisのデータセットもscikit-learnに入っている）

主成分分析を行うコードは

pca = PCA(n_components = 2)　→2次元に圧縮

pca.fit(x)

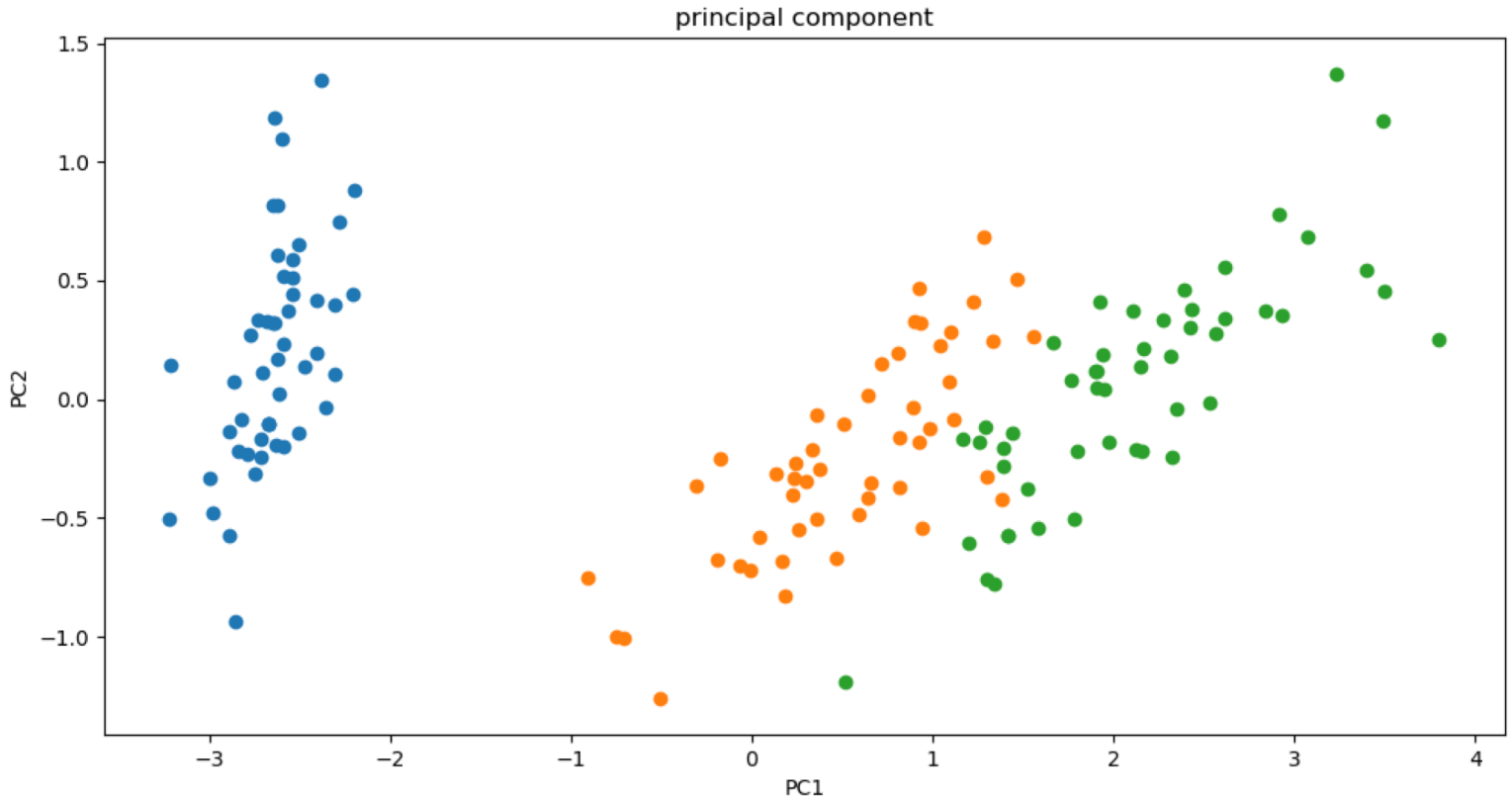
たったこれだけ

主成分分析

②ライブラリmatplotlibを使って視覚化

matplotlibのなかでもpyplotというものを使ってプロット
コードは
pyplot.scatter(x,y)
引数xとyは出力するデータを入れます

主成分分析 –結果-



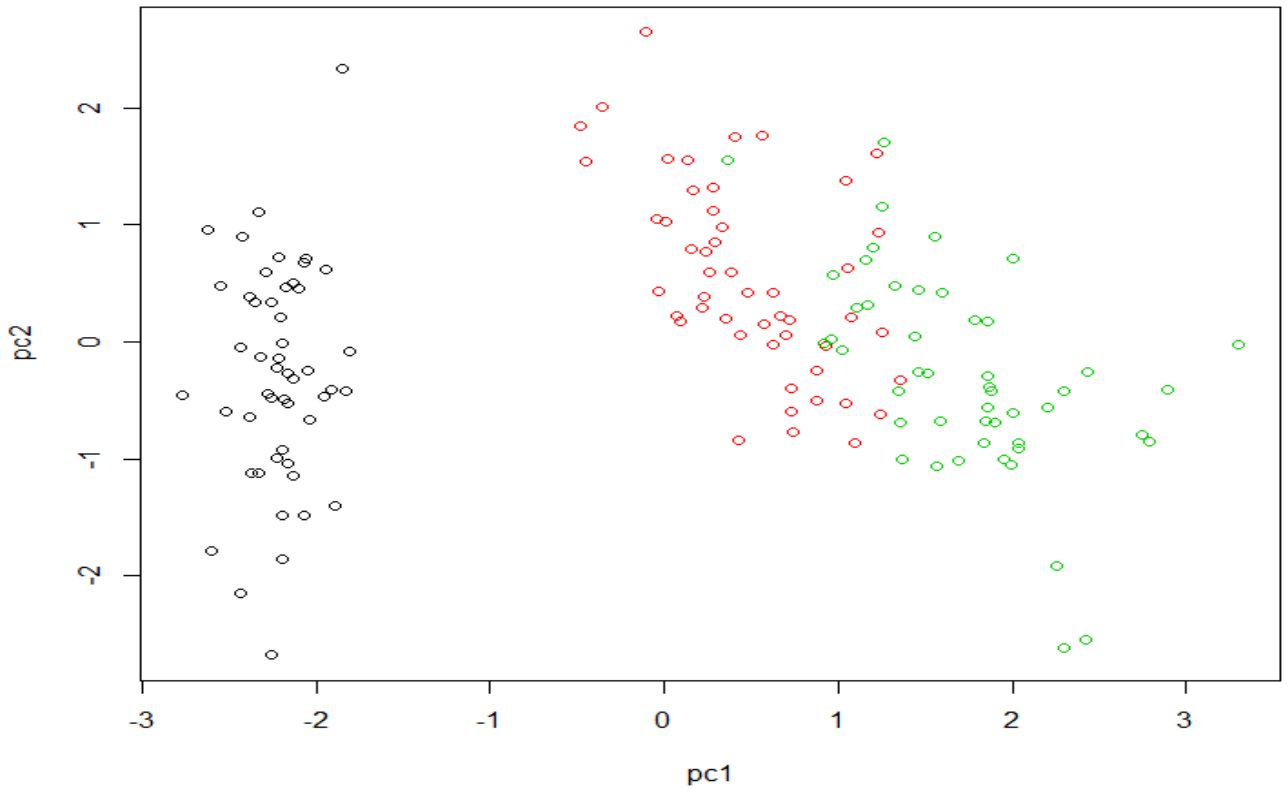
グラフ1つにデータをプロットすることができ、最初の6つのグラフよりもだいぶ見やすくなった！！

続いて

せっかくなのでRでも主成分分析

　Rでの主成分分析の詳しい方法は輪講でやった（つもり）なので割愛します

主成分分析 –結果-

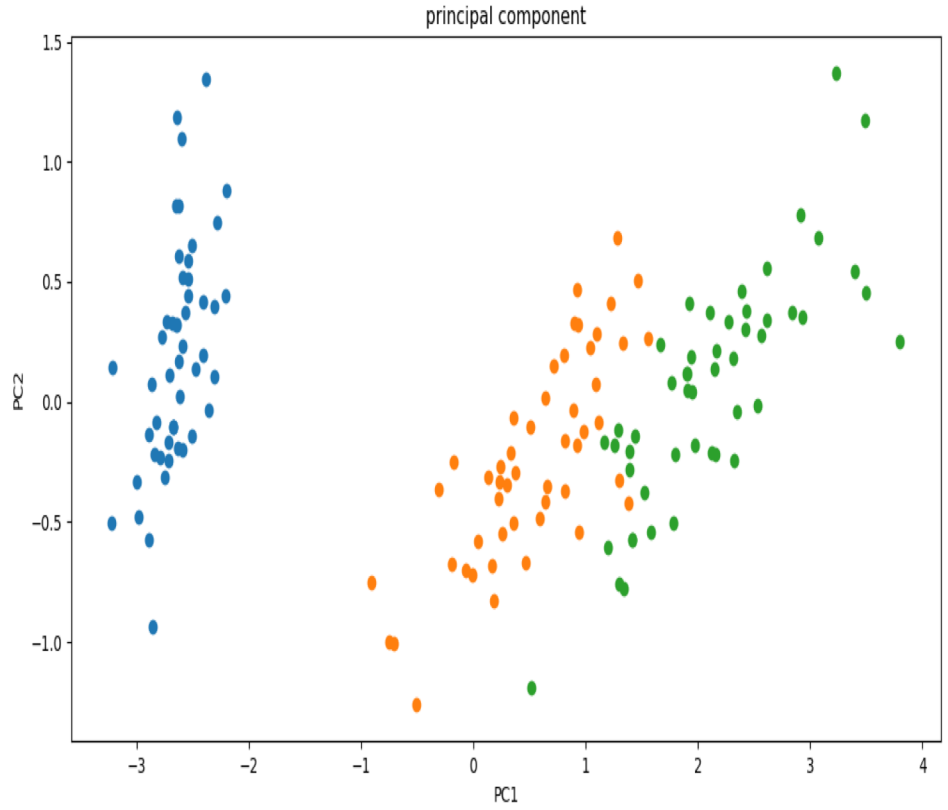


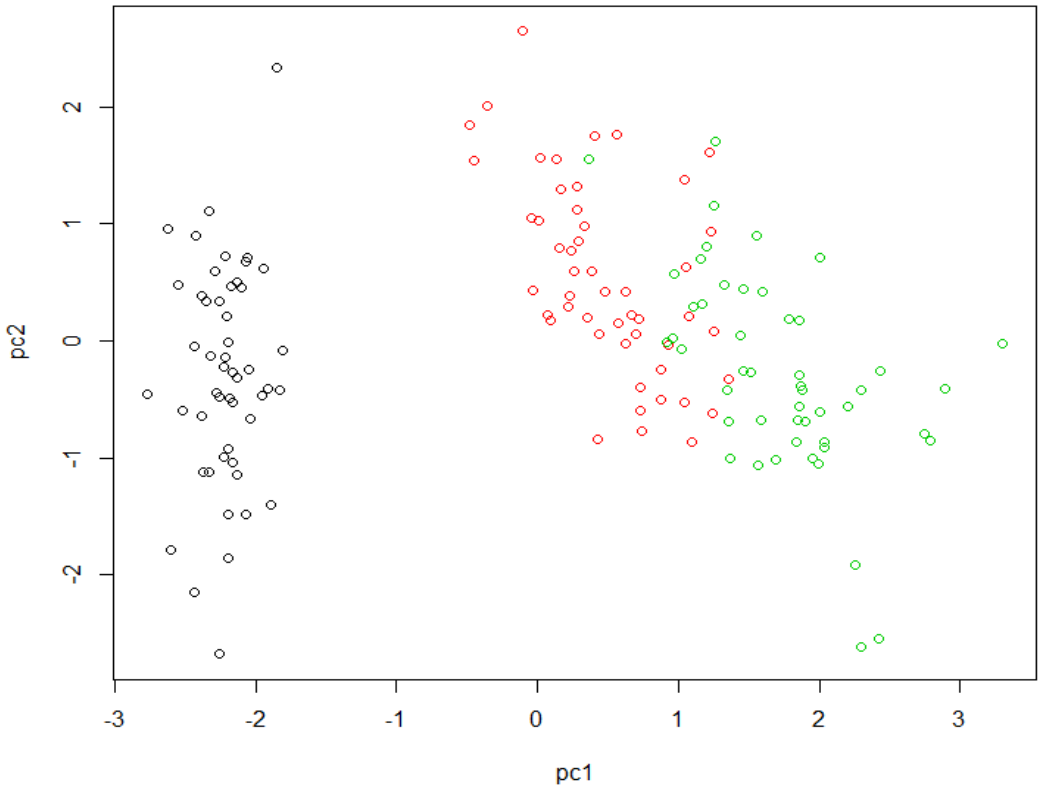
RとPythonの比較

主成分分析についてRとPythonを比較してみます

RとPythonの比較　–結果-

微妙に結果が違う…
色々調べてみましたが原因はわからず終いでした





RとPythonの比較　-コード量-

R

```r
#irisの元のデータを主成分分析して視覚化

#品種名を削除
data <- iris[1:4]

#主成分分析
iris.pca<-prcomp(data, scale=TRUE)
pc1 <- iris.pca$x[,1]
pc2 <- iris.pca$x[,2]

#結果をプロット
#各品種はマーカーの色を変える
label <- as.factor(iris[,5])
plot(pc1, pc2, col=label)
```

Python

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#irisの元のデータを主成分分析して視覚化

import numpy as np
from matplotlib import pyplot as plt
from sklearn.decomposition import PCA
from sklearn import datasets

def main():
    iris = datasets.load_iris()

    #特徴量の入ったデータ（4次元）
    x = iris.data
    #データと品種の対応
    targets = iris.target

    #グラフ描画サイズを設定する
    plt.figure(figsize=(12, 6))

    #主成分分析
    pca = PCA(n_components = 2)
    pca.fit(x)
    x_pca = pca.transform(x)


    #結果をプロット
    #各品種はマーカーの色を変える
    for label in np.unique(targets):
        plt.scatter(x_pca[targets == label, 0],
                    x_pca[targets == label, 1])
        plt.title('principal component')
        plt.ylabel('PC2')
        plt.xlabel('PC1')
        plt.autoscale()

    plt.show()

if __name__ == '__main__':
    main()
```

RとPythonの比較　-コード量-

明らかにRの方が少ない
→やはり統計関連を扱うにはRの方が向いている

感想

- コードの量からもわかるようにライブラリのインポートなどがあって正直Rの方が簡単に主成分分析できた
- 主成分分析を視覚化した際にRとPythonで結果が異なった点に関しては色々調べてみたが結局わからなかった
- RとPythonの比較ということで1年通して菊池研で学んだことをこのプレゼンで発揮できた（気がする）