FUNDAMENTALS OF MACHINE LEARNING IN DATA SCIENCE

CSIS 3290

DIMENSION REDUCTION

PRINCIPAL COMPONENT ANALYSIS (PCA)

FATEMEH AHMADI

Principal Component Analysis (PCA)

- Principal Component Analysis (PCA) is a **linear dimensionality reduction** technique that can be utilized for extracting information from a high-dimensional space by projecting it into a lower-dimensional sub-space. It tries to preserve the essential parts that have more variation of the data and remove the non-essential parts with fewer variation.
- Dimensions are nothing but features that represent the data. PCA is a **statistical** procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) **into a set of values of linearly uncorrelated variables called principal components**.

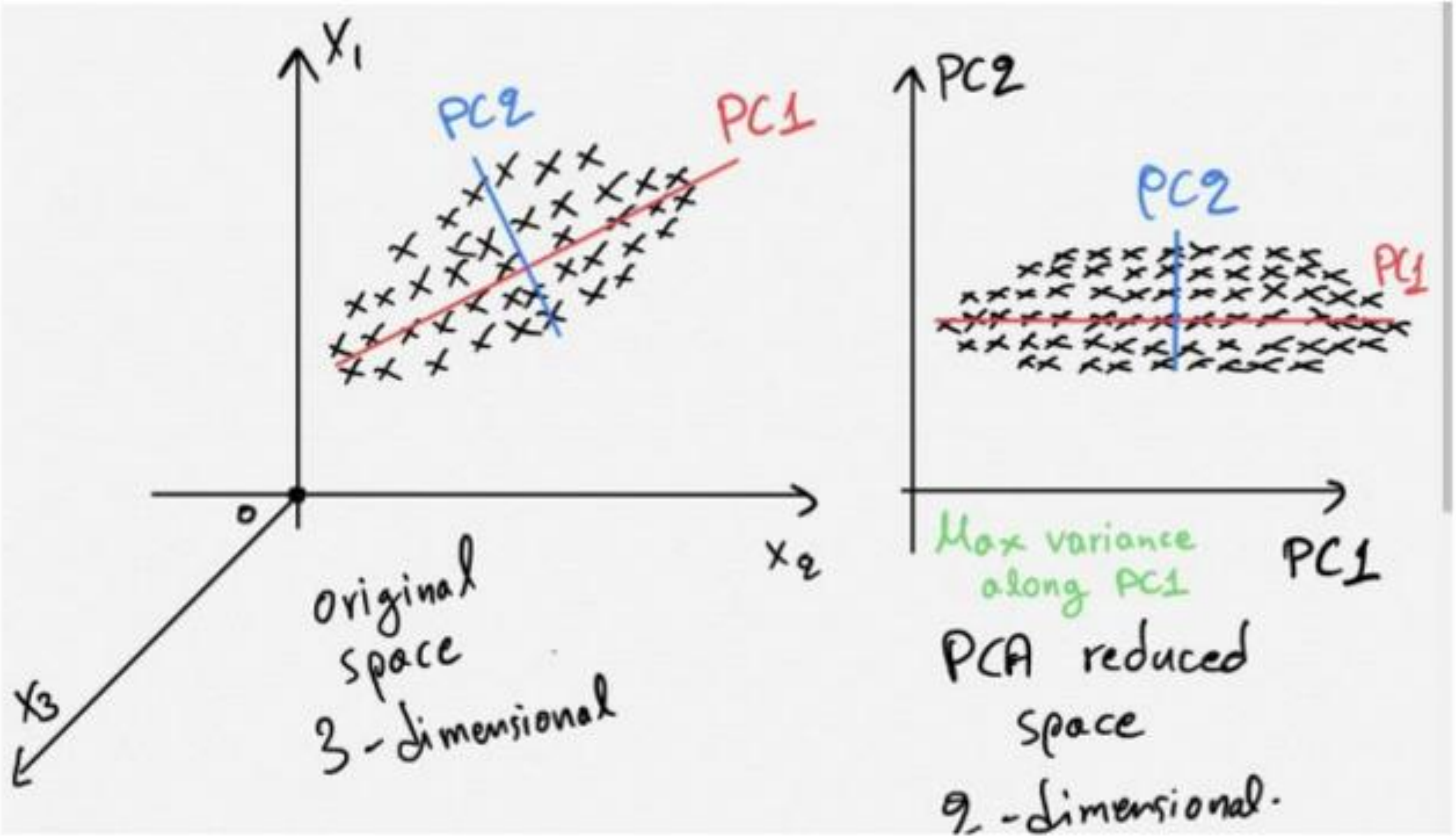
https://www.datacamp.com/tutorial/principal-component-analysis-in-python#!#SnippetTab

PCA

When/Why to use PCA

- PCA technique is particularly useful in processing data where **multi-colinearity** exists between the **features/variables**.
- PCA can be used when **the dimensions of the input features are high** (e.g. a lot of variables).
- PCA can be also used for **denoising** and **data compression**.

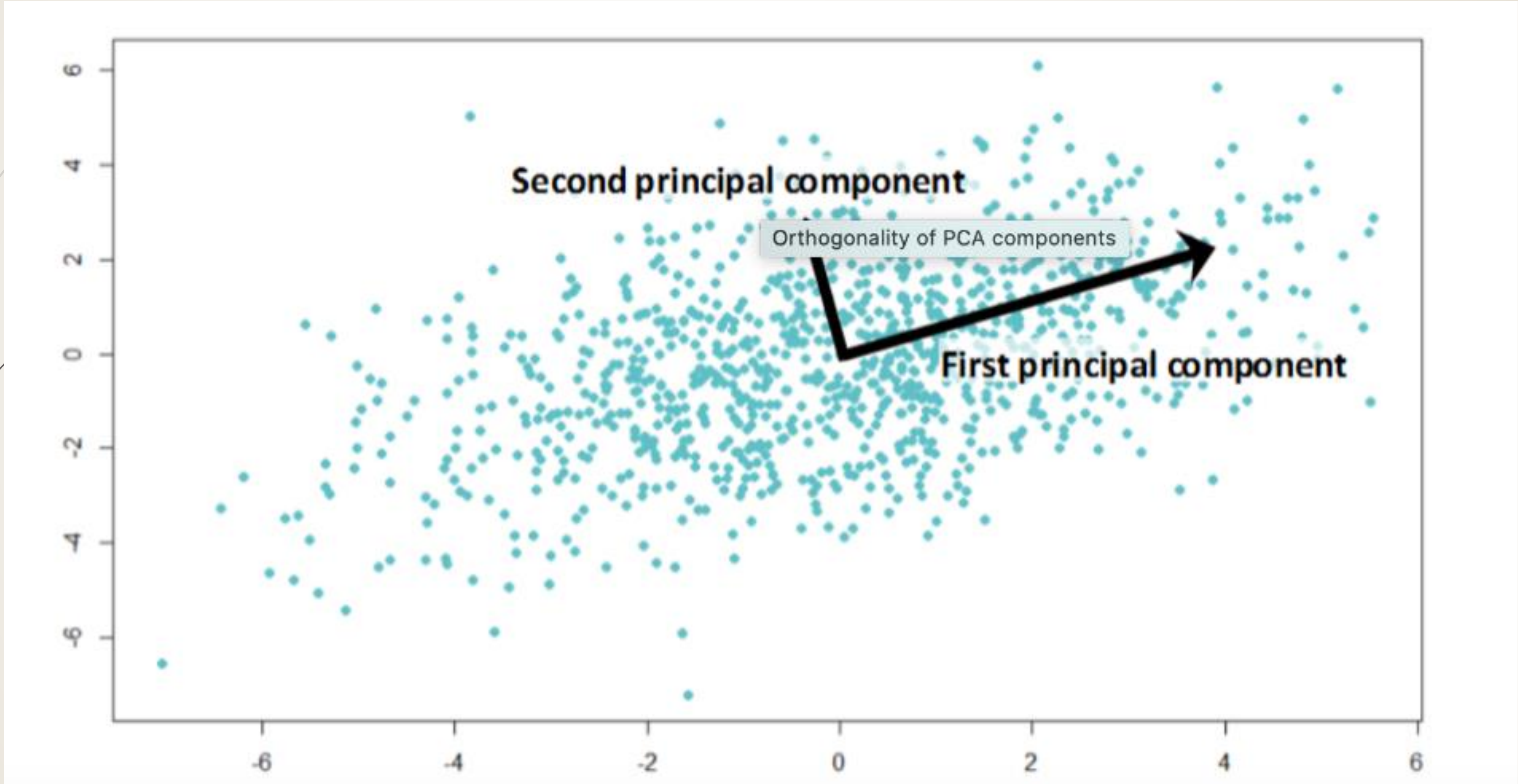
PCA



https://towardsdatascience.com/pca-clearly-explained-how-when-why-to-use-it-and-feature-importance-a-guide-in-python-7c274582c37e

PCA

https://www.bing.com/images/search?view=detailV2&ccid=77HVFI5L&id=D2DF62D01D8BC2EF468A4F84E7425B3E6103AEB6&thid=OIP.77HVFI5LCgFrE6qF1WmC0gHaD3&mediaurl=https%3a%2f%2fmiro.medium.com%2fmax%2f3264%2f1*2S1CN1w6FtbAIkHOb1fcsg.png&cdnurl=https%3a%2f%2fth.bing.com%2fth%2fid%2fR.efb1d5148e4b0a016b13aa85d56982d2%3frik%3dtq4DYT5bQueETw%26pid%3dImgRaw%26r%3d0&exph=852&expw=1632&q=principal+component+analysis&simid=608013231911107578&FORM=IRPRST&ck=008DC95F18D5C7D31F3E9A19F8EE5B72&selectedIndex=43&ajaxhist=0&ajaxserp=0



PCA in Python

```
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
```

```
In [33]: iris1=load_iris()

In [34]: iris2=iris1.data[:,[0,2]]

In [36]: pca1=PCA()

In [37]: pca1.fit(iris2)

Out[37]: ▾ PCA
         PCA()
```

PCA in Python

First and second components (by default <u>two</u> components will be built)

```
In [84]: pc=pca1.components_[0]
         pcc=pca1.components_[1]
```

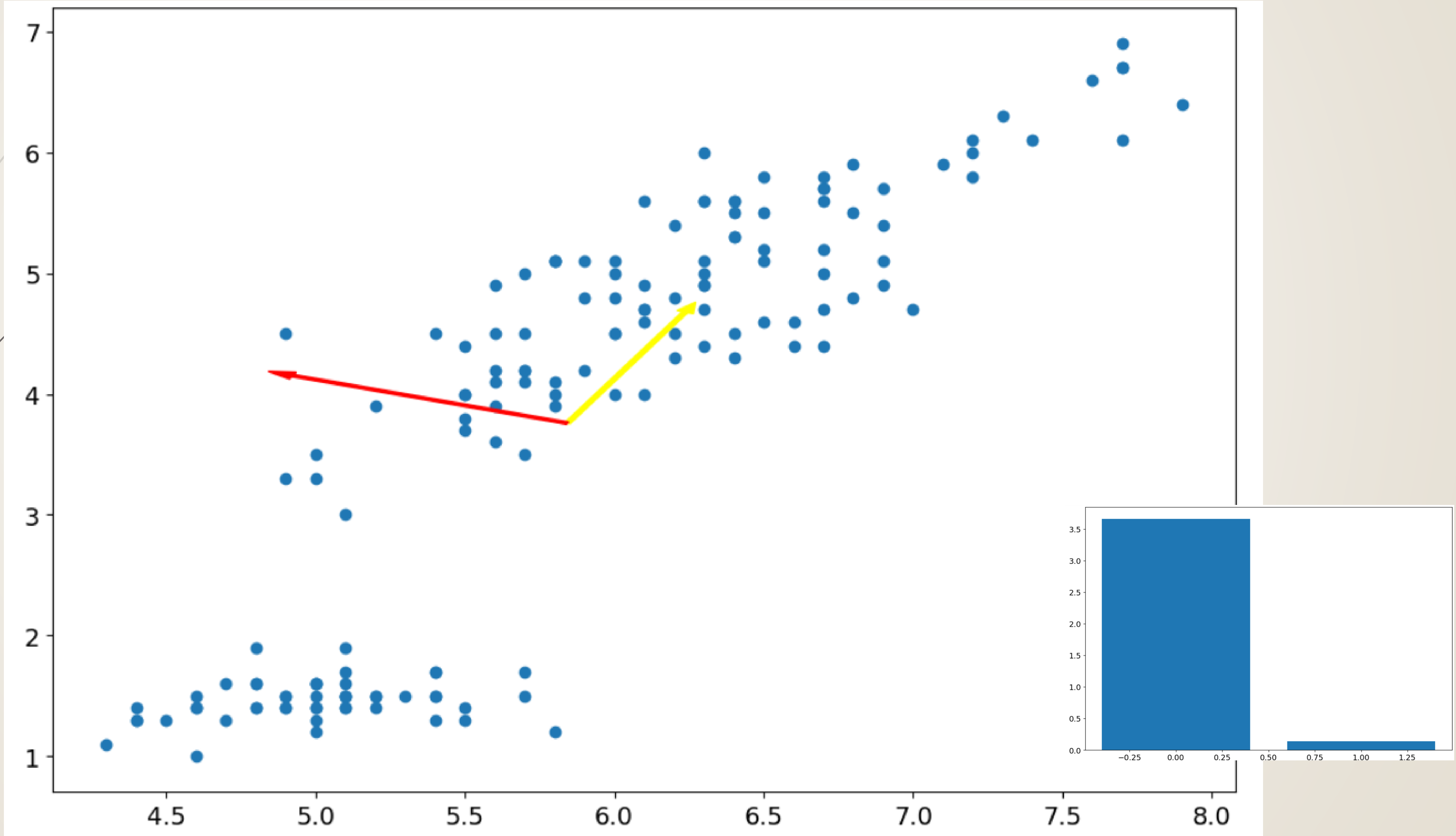
```
In [85]: mean1=pca1.mean_
```

```
In [87]: plt.scatter(iris2[:,0],iris2[:,1])
         plt.arrow(mean1[0],mean1[1],pc[0],pc[1],color='yellow',width=0.02)
         plt.arrow(mean1[0],mean1[1],pcc[0],pcc[1],color='red',width=0.02)
         plt.show()
```

x and y of the start point of the arrow

x and y of the end point of the arrow

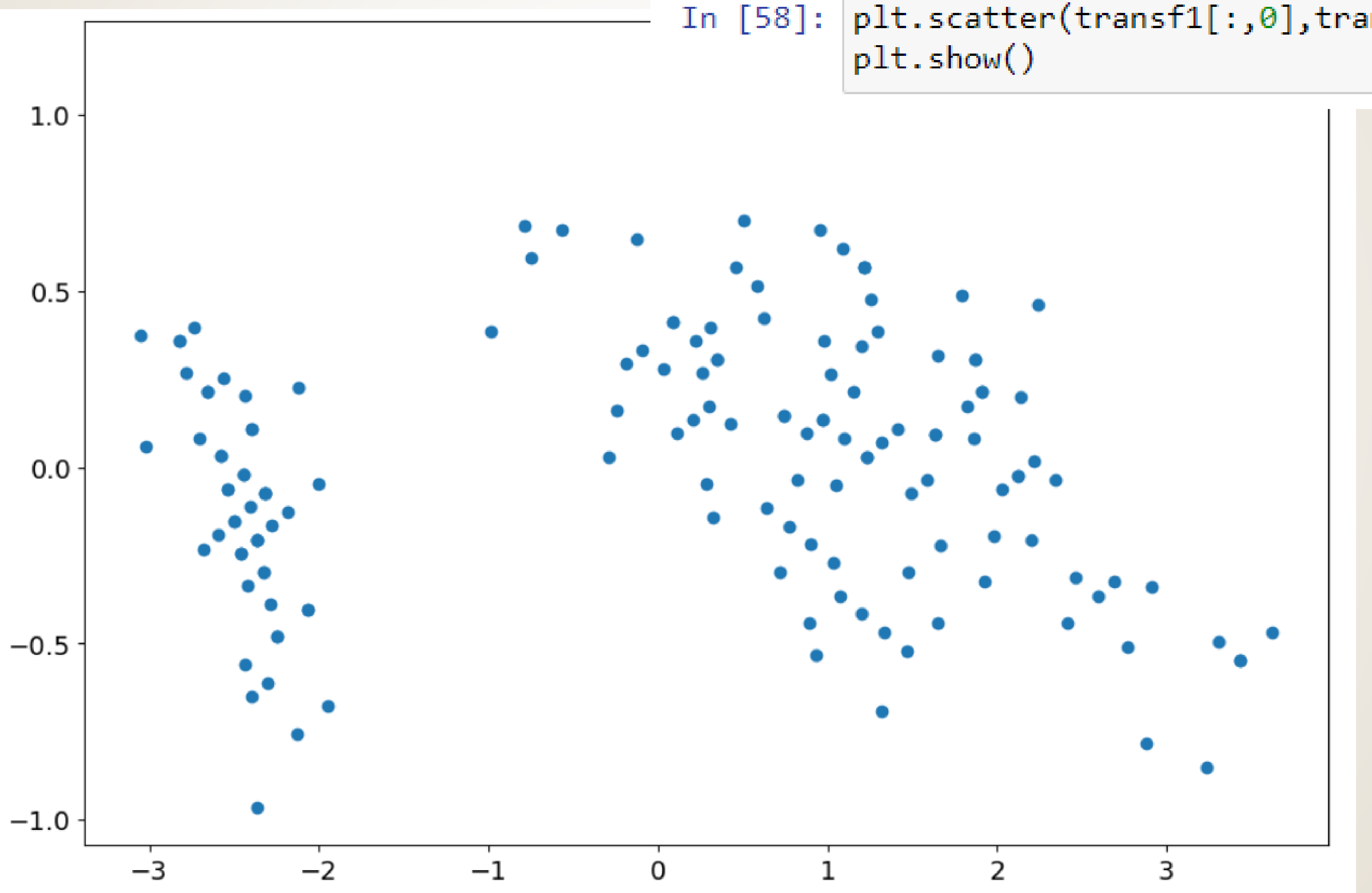
PCA in Python



PCA in Python

In [56]:
```
transf1=pca1.transform(iris2)
```

In [58]:
```
plt.scatter(transf1[:,0],transf1[:,1])
plt.show()
```



PCA in Python

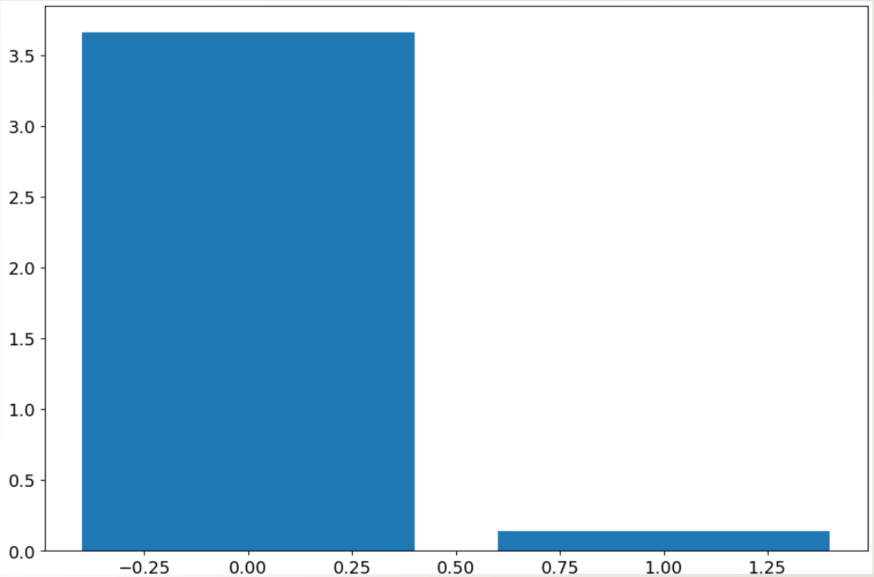
```
In [67]: pca1.n_components_
Out[67]: 2

In [68]: nf=range(pca1.n_components_)

In [69]: var_nf=pca1.explained_variance_

In [70]: var_nf
Out[70]: array([3.66189877, 0.1400726 ])

In [71]: plt.bar(nf,var_nf)
         plt.show()
```




```
In [59]: pca1.explained_variance_ratio_
Out[59]: array([0.9631579, 0.0368421])
```

PCA in Python

```
In [72]: pca2=PCA(n_components=2)

In [74]: pca2.fit(iris1.data)
Out[74]:
▾ PCA
PCA(n_components=2)

In [76]: transf3=pca2.transform(iris1.data)
         transf3
```

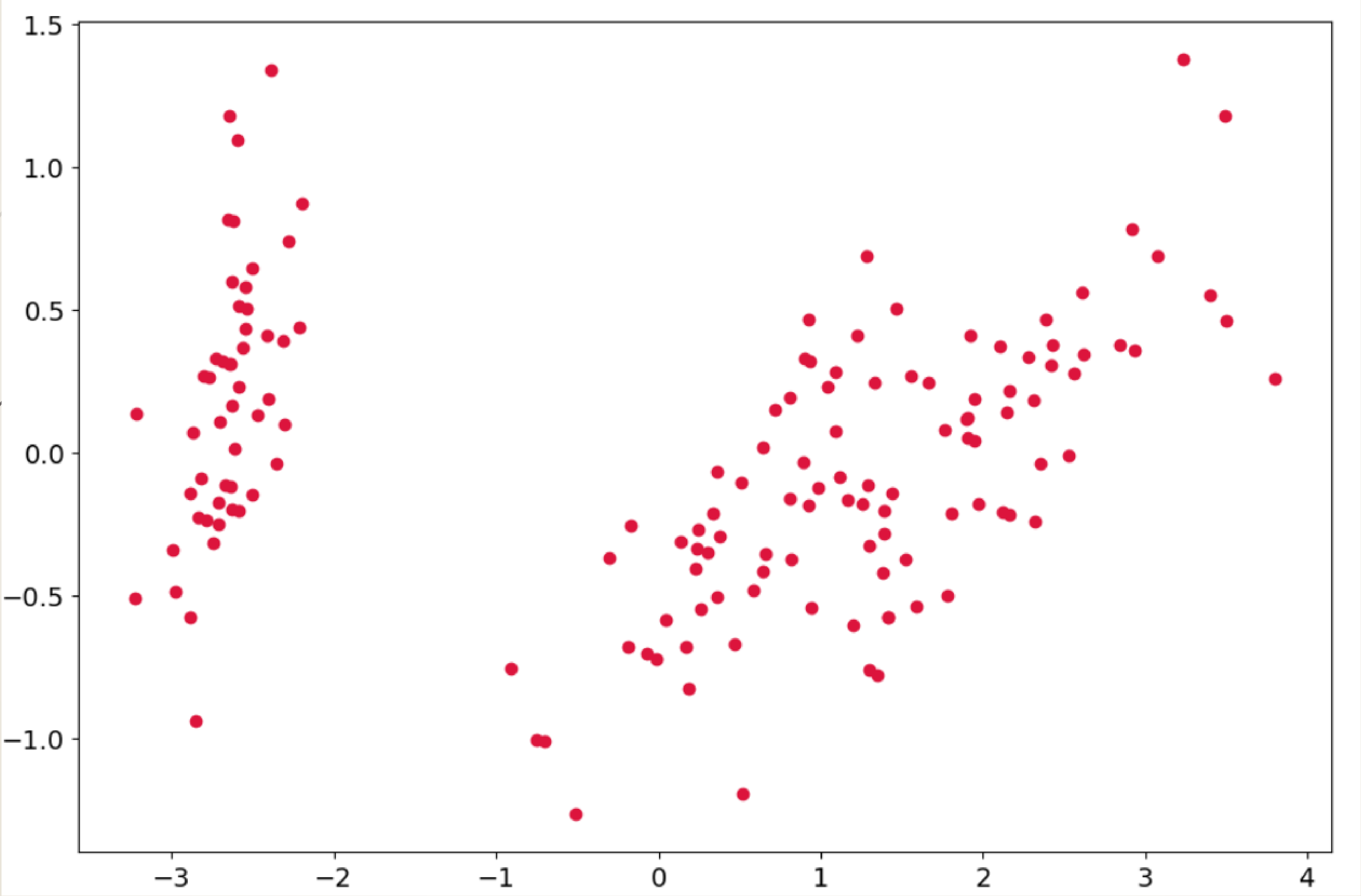
PCA in Python

As a Feature Reduction Approach

```
Out[76]: array([[-2.68412563,  0.31939725],
               [-2.71414169, -0.17700123],
               [-2.88899057, -0.14494943],
               [-2.74534286, -0.31829898],
               [-2.72871654,  0.32675451],
               [-2.28085963,  0.74133045],
               [-2.82053775, -0.08946138],
               [-2.62614497,  0.16338496],
               [-2.88638273, -0.57831175],
               [-2.6727558 , -0.11377425],
               [-2.50694709,  0.6450689 ],
               [-2.61275523,  0.01472994],
               [-2.78610927, -0.235112  ],
               [-3.22380374, -0.51139459],
               [-2.64475039,  1.17876464],
               [-2.38603903,  1.33806233],
               [-2.62352788,  0.81067951],
               [-2.64829671,  0.31184914],
               [-2.19982032,  0.87283904],
```

PCA in Python


```
In [79]: plt.scatter(transf3[:,0],transf3[:,1], c='crimson')
         plt.show()
```



Reference

Data Mining, Concepts and Techniques, Jiawei Han, Micheline Kamber, Jian Pei. MK. Chapter 9.