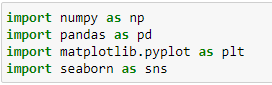
Programming for AI Lab 11

|  |  |
| --- | --- |
| Course: Programming for AI (AI2001) | Semester: Fall 2023 |
| Instructor: Sameer Faisal | T.A: N/A |
| Note:   * Maintain discipline during the lab. * Listen and follow the instructions as they are given. * Just raise hand if you have any problem. * Completing all tasks of each lab is compulsory. * Get your lab checked at the end of the session. |  |

**Principal Component Analysis (PCA)**

Principal Component Analysis (PCA) is a dimensionality reduction technique used in statistics and machine learning. It identifies and focuses on the most important features or components of a dataset, simplifying its complexity while retaining key information.



**Dataset:**

**Breast cancer wisconsin (diagnostic) dataset**

**Data Set Characteristics:**

* Number of Instances: 569
* Number of Attributes: 30 numeric, predictive attributes and the class
* Attribute Information:

- radius (mean of distances from center to points on the perimeter)

- texture (standard deviation of gray-scale values)

- perimeter

- area

- smoothness (local variation in radius lengths)

- compactness (perimeter^2 / area - 1.0)

- concavity (severity of concave portions of the contour)

- concave points (number of concave portions of the contour)

- symmetry

- fractal dimension ("coastline approximation" - 1)

The mean, standard error, and "worst" or largest (mean of the three

worst/largest values) of these features were computed for each image,

resulting in 30 features. For instance, field 0 is Mean Radius, field

10 is Radius SE, field 20 is Worst Radius.

- class:

- WDBC-Malignant

- WDBC-Benign

Summary Statistics:

===================================== ====== ======

Min Max

===================================== ====== ======

radius (mean): 6.981 28.11

texture (mean): 9.71 39.28

perimeter (mean): 43.79 188.5

area (mean): 143.5 2501.0

smoothness (mean): 0.053 0.163

compactness (mean): 0.019 0.345

concavity (mean): 0.0 0.427

concave points (mean): 0.0 0.201

symmetry (mean): 0.106 0.304

fractal dimension (mean): 0.05 0.097

radius (standard error): 0.112 2.873

texture (standard error): 0.36 4.885

perimeter (standard error): 0.757 21.98

area (standard error): 6.802 542.2

smoothness (standard error): 0.002 0.031

compactness (standard error): 0.002 0.135

concavity (standard error): 0.0 0.396

concave points (standard error): 0.0 0.053

symmetry (standard error): 0.008 0.079

fractal dimension (standard error): 0.001 0.03

radius (worst): 7.93 36.04

texture (worst): 12.02 49.54

perimeter (worst): 50.41 251.2

area (worst): 185.2 4254.0

smoothness (worst): 0.071 0.223

compactness (worst): 0.027 1.058

concavity (worst): 0.0 1.252

concave points (worst): 0.0 0.291

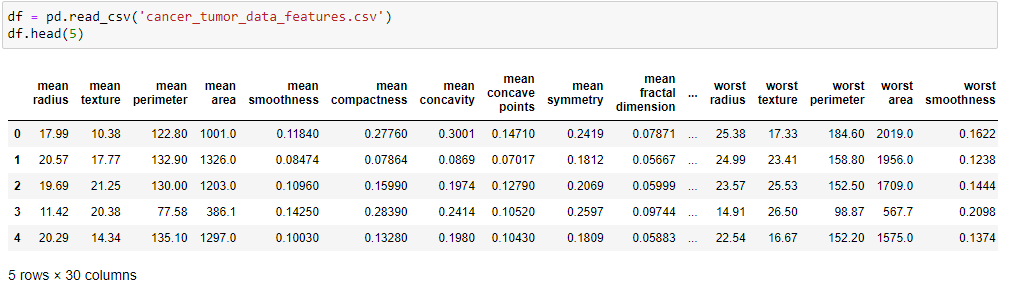
symmetry (worst): 0.156 0.664

fractal dimension (worst): 0.055 0.208

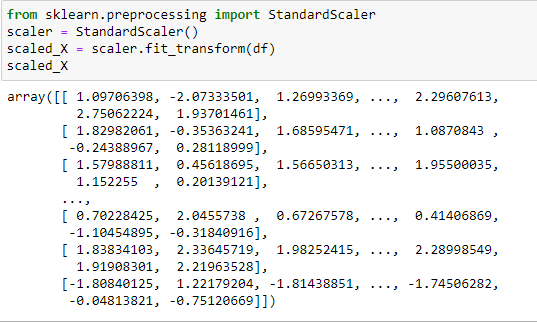
===================================== ====== ======

Missing Attributes: none

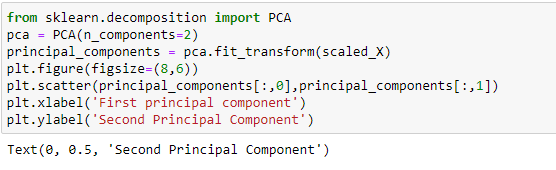
Class Distribution: 212 - Malignant, 357 – Benign

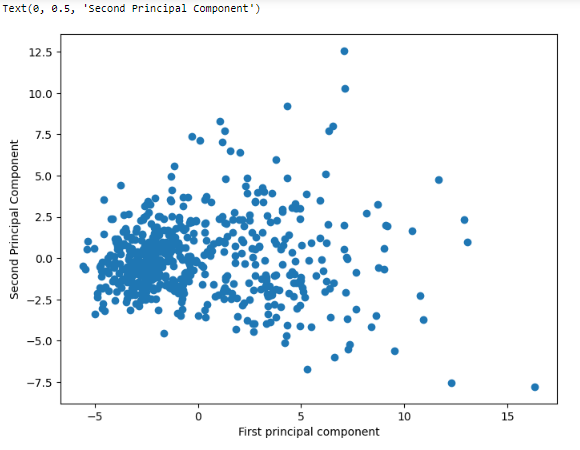


**Scaling the Data:**

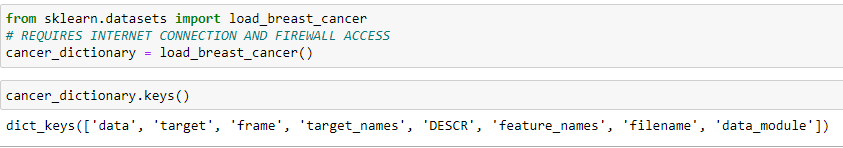


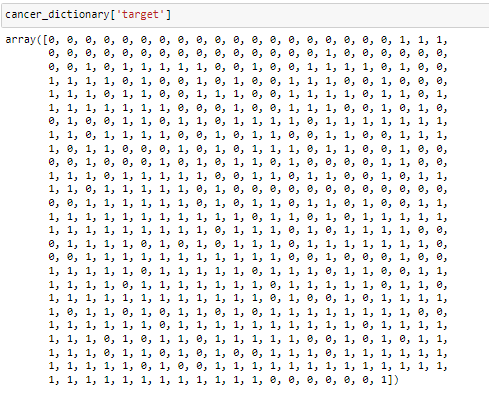
**Scikit - Learn Implementation:**

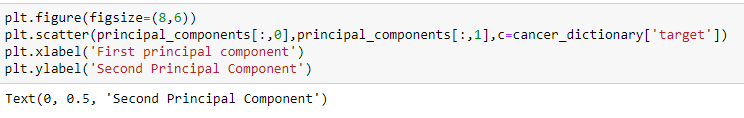
****

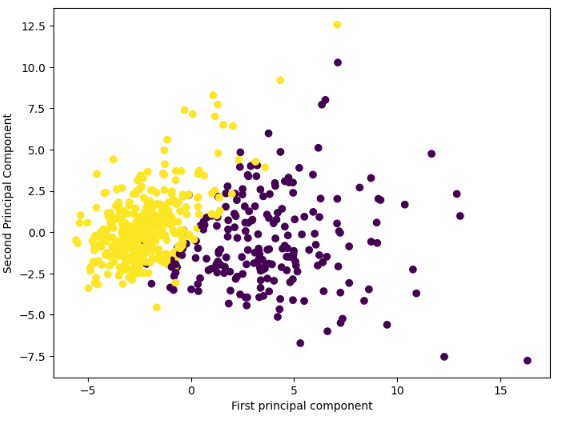
****

**Loading the Dataset:**

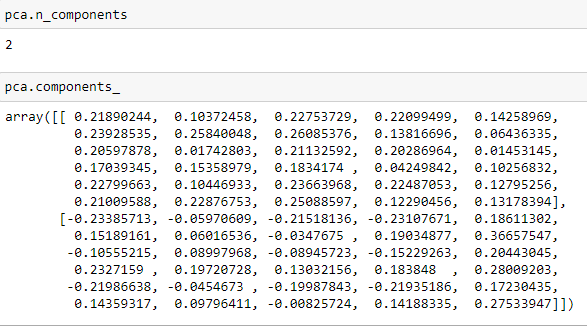
****

****

****

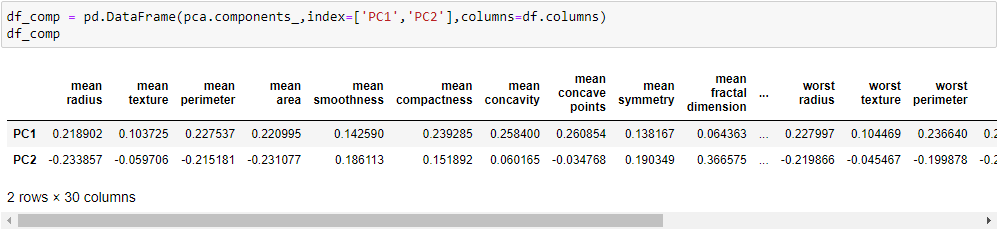
****

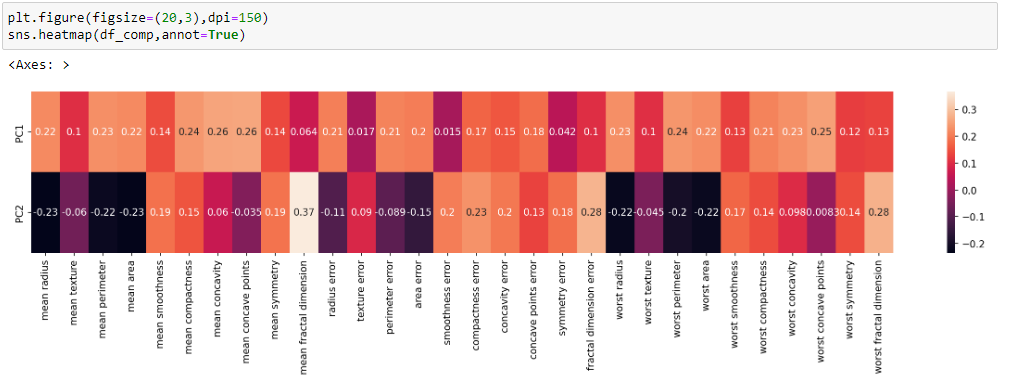
**Fitted Model Attributes:**

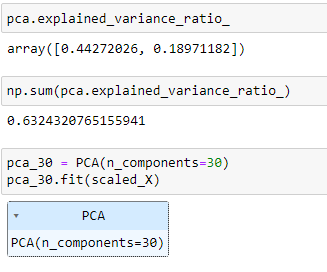
****

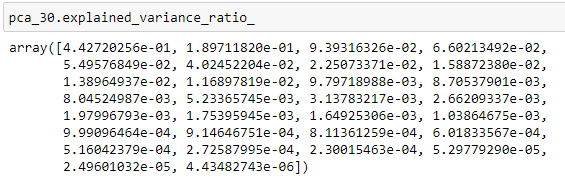
In this numpy matrix array, each row represents a principal component, Principal axes in feature space, representing the directions of maximum variance in the data. The components are sorted by explained\_variance\_.

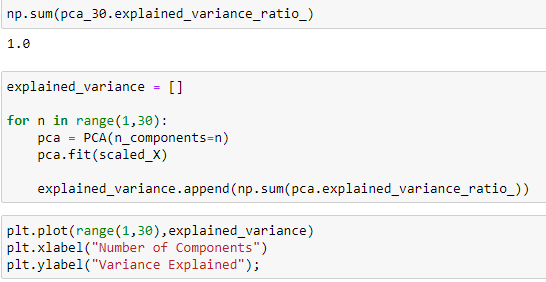
We can visualize this relationship with a heatmap:

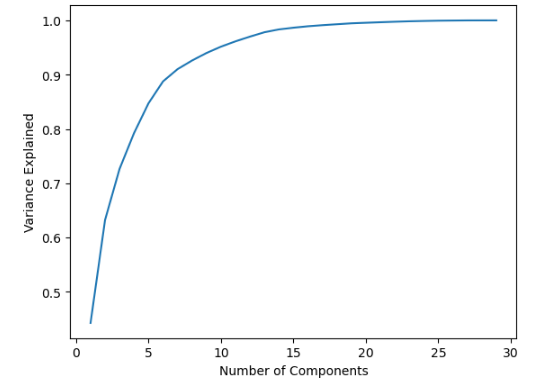












**Lab Tasks**

1. Figure out which handwritten digits are most differentiated with PCA:

Imagine you are working on an image recognition service for a postal service. It would be very useful to be able to read in the digits automatically, even if they are handwritten. (Quick note, this is very much how modern postal services work for a long time now and its actually more accurate than a human). The manager of the postal service wants to know which handwritten numbers are the hardest to tell apart, so he can focus on getting more labeled examples of that data. You will have a dataset of hand written digits (a very famous data set) and you will perform PCA to get better insight into which numbers are easily separable from the rest.

Complete the following tasks:

* Import the libraries and relevant data set.
* Create a new DataFrame called pixels that consists only of the pixel feature values by dropping the number\_label column.
* Grab a single image row representation by getting the first row of the pixels DataFrame.
* Convert the above single row Series into a numpy array.
* Reshape this numpy array into an (8,8) array.
* Use Matplotlib or Seaborn to display the array as an image representation of the number drawn. Remember your palette or cmap choice would change the colors, but not the actual pixel values.
* Use Scikit-Learn to scale the pixel feature dataframe.
* Perform PCA on the scaled pixel data set with 2 components.
* Show how much variance is explained by 2 principal components.
* Create a scatterplot of the digits in the 2-dimensional PCA space, color/label based on the original number\_label column in the original dataset.