Principal Component Analysis (PCA)

Presented by Farzad Mozafari
Instructor Dr. Yasser Alginahi

Date November 10, 2023

Outline of the Presentation

- Introduction
- Objectives of PCA
- Pseudocode of PCA
- Terminology
- Algorithm
- PCA with Python
- Applications
- Conclusion
- References

Introduction (feature reduction)

- Feature reduction refers to the mapping of the original high-dimensional data onto a lower-dimensional space.[6]
  - Criterion for feature reduction [3] can be different based on different problem settings.
    - **Unsupervised setting:** Minimize the information loss.
    - **Supervised setting:** Maximize the class discrimination.
- Why feature reduction?
- **Visualization:** Projection of high-dimensional data onto 2D or 3D.[2]
  - Most machine learning and data mining techniques may not be effective for high-dimensional data [5].
  - Query accuracy and efficiency degrade rapidly as the dimension increases.
- **Data compression:** Efficient storage and retrieval.
- **Noise removal:** Positive effect on query accuracy.

Feature reduction algorithms

- **Unsupervised**
  - Latent Semantic Indexing (LSI): truncated SVD
  - Independent Component Analysis (ICA)
  - Principal Component Analysis (PCA) [1]
  - Canonical Correlation Analysis (CCA)
- **Supervised**
  - Linear Discriminant Analysis (LDA)
- **Semi-supervised**
  - Research topic
- **Application of feature reduction**
  - Reduce number of dimensions in data.
  - Find patterns in high-dimensional data.
  - Visualize data of high dimensionality.

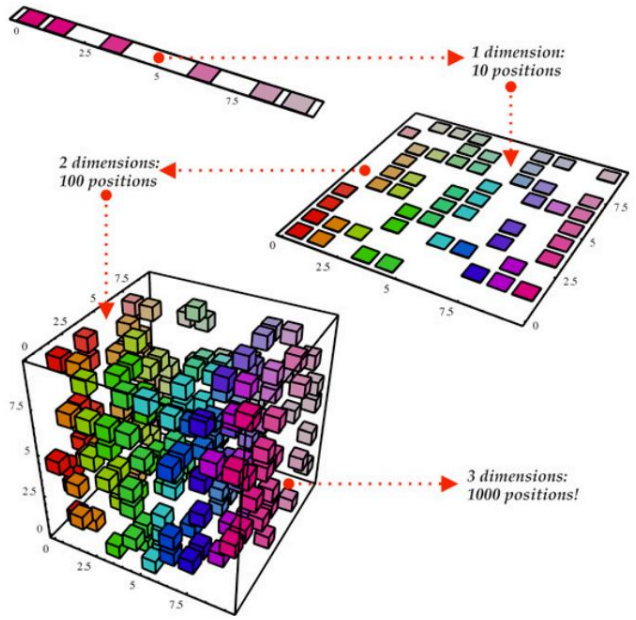


Fig. 1. reduction of features from 3 to 1.
https://tinyurl.com/ybhspnry [7]

Principal Component Analysis (PCA)

- **Principal Component Analysis (PCA)**

  - Reduce the dimensionality of a data set by finding a new set of variables, smaller than the original set of variables
  - Retains most of the sample's information.
  - Useful for the compression and classification of data.

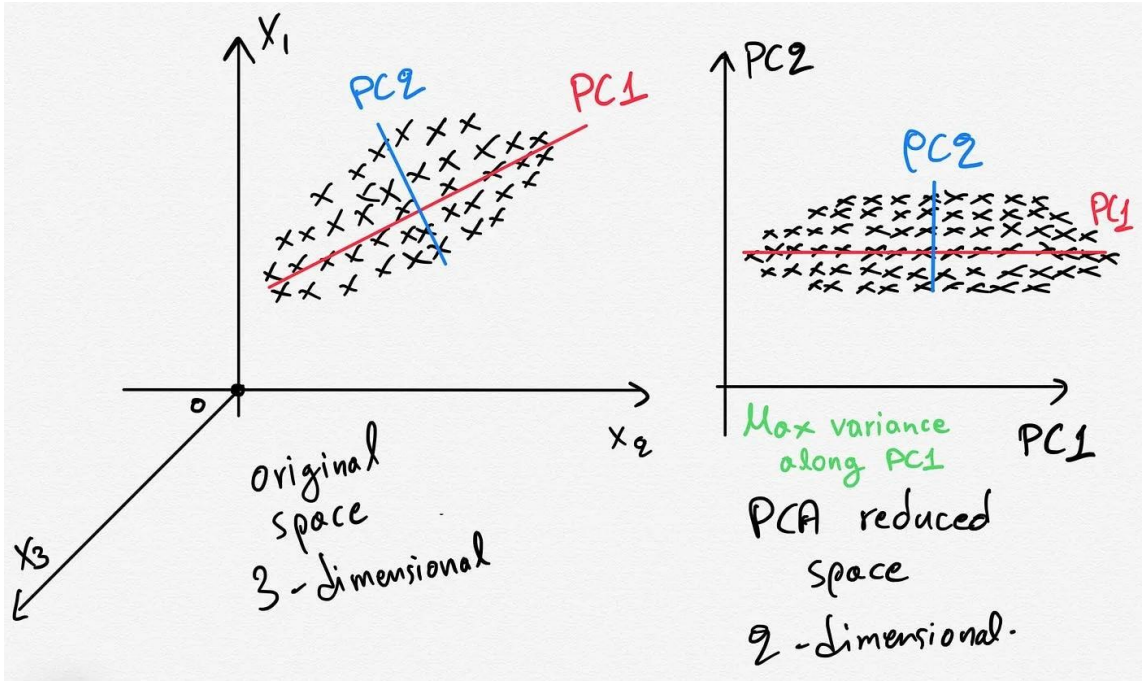


Fig. 2: Transformation of the data from 3D feature space to 2d feature space. https://tinyurl.com/yfhywe27 [8]

Objectives of PCA

- **Principal Component Analysis** is a technique used to:
  - Reduce the dimensionality of the data set
  - Identify new meaningful underlying variables
  - Loose minimum information
- **N dimensional data** is taken, and **M orthogonal directio** are found.
  - ✓ It means that the data has the most variance.
  - ✓ These principal directions of M form a lower dimensional subspace.
- An **N-dimensional data** point can be represented by its predictions in the **main directions of M.**

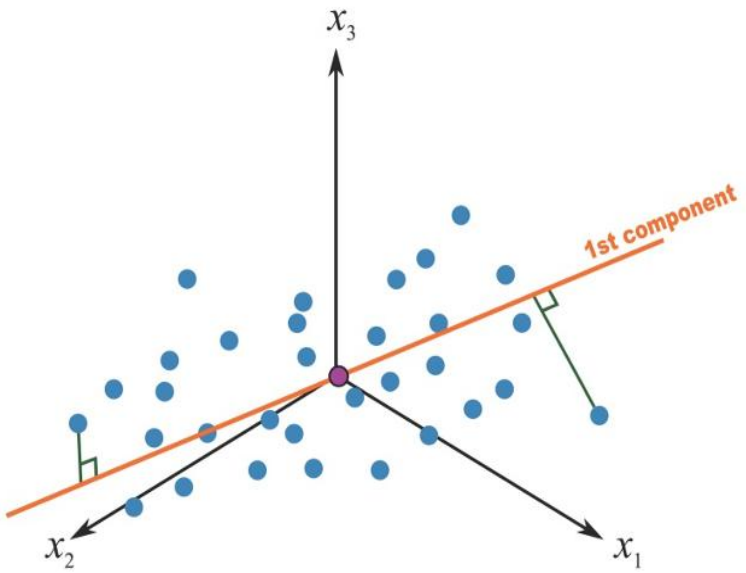


Fig. 3. First principal component in a 3-covariates setting. https://tinyurl.com/yefpr7uh [11]

Pseudocode of PCA

- **Standardization**
  - ✓ Importance of data standardization with mean and standard deviation.[2]
  - ✓ PCA relies on the assumption of normal distribution of data and is very sensitive to the characteristic or variance of the variable.[4, 9]
- **Compute the covariance matrix**
  - ✓ Covariance matrix, a square matrix containing data variance and variable covariance [1].
  - ✓ Quantifying the relationship between variables through a covariance matrix [3].
    - ❖ Basically, providing an empirical picture of the data [1] and revealing the correlations between the characteristics of the data.
- **Compute eigenvectors and eigenvalues from the covariance matrix**
  - ✓ Calculation of eigenvectors and eigenvalues to identify the principal components that record the most variance of the data.
  - ✓ Showing the direction of data dispersion or the highest variance with eigenvectors [3].
    - ❖ Relative importance of directions with eigenvalues [3].
- **Compute the feature vector and principal components**
  - ✓ The eigenvectors act as principal components, while the eigenvalues determine their relative importance.
  - ✓ It is noteworthy that all eigenvectors are perpendicular to the vector before them [10].
- **Project the data onto the selected principal components for dimensionality reduction**
  - ✓ Considering the two main components out of the four components, reduce the dimensions of the data using the following formula [4].
  - ✓ ***Final Data Set= Standardized Original Data Set * Feature Vector***

Terminology

- Variance
- Covariance
- Eigenvectors & Eigenvalues
- Principal Components

Terminology (Variance)

- Standard deviation:
  - Average distance from mean to a point
- Variance:
  - Standard deviation squared
  - One-dimensional measure

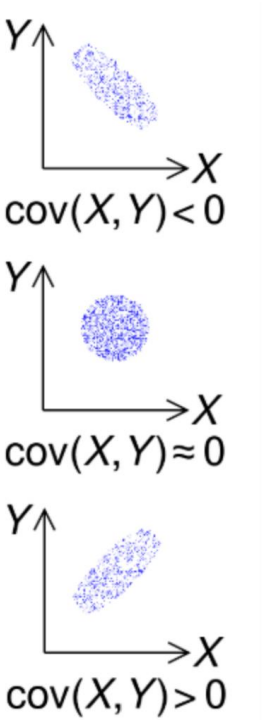
$$s^2 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{(n-1)} \qquad (1)$$

Terminology (Covariance)

- How *two* dimensions vary from the mean with respect to each other

$$\text{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)} \quad (2)$$

- $\text{cov}(X,Y) > 0$: Dimensions increase together
- $\text{cov}(X,Y) < 0$: One increases, one decreases
- $\text{cov}(X,Y) = 0$: Dimensions are independent



https://tinyurl.com/ehvj84x7

Terminology (Eigenvalues & Eigenvectors)

- **Eigenvalues** measure the amount of the variation explained by each PC (largest for the first PC and smaller for the subsequent PCs).
  - The extracted uncorrelated components are called **Principal Components (PC).**
- **Eigenvectors** provides the weights to compute the uncorrelated PC. These vectors give the directions in which the data cloud is stretched most.

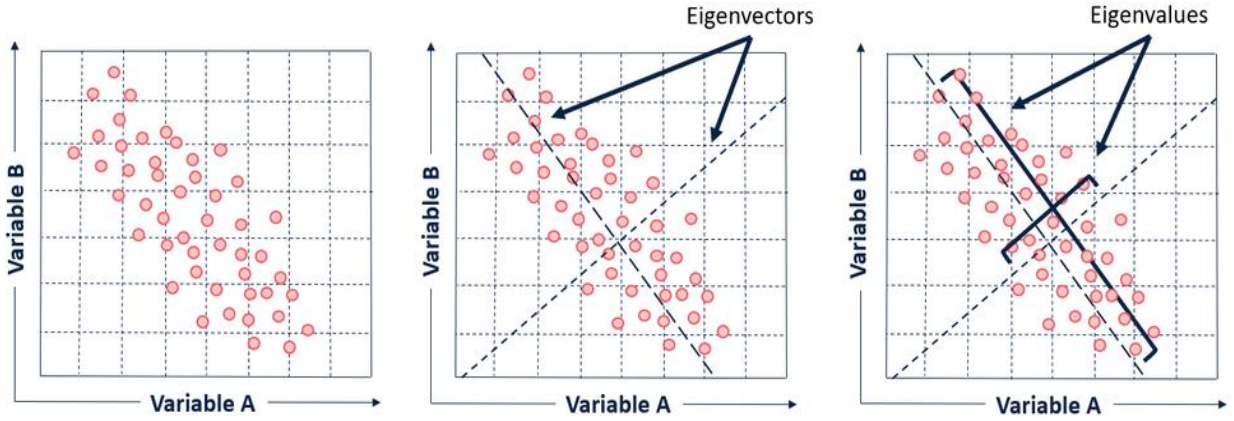


Fig 4. Eigenvectors and eigenvalues of the dataset with two variables A & B taken from https://tinyurl.com/5czfny39 [10].

Terminology (cont'd)

- Vectors **x** having same direction as $A\mathbf{x}$ are called *eigenvectors* of A

  (A is an n-by-n matrix).

- In the equation $A\mathbf{x}=\lambda\mathbf{x}$, λ is called an *eigenvalue* of A. [4]

$$\mathbf{Ax=\lambda x \Leftrightarrow (A-\lambda I)x=0} \qquad (3)$$

- How to calculate **x** and λ:
  - Calculate $det(A-\lambda I)$, yields a polynomial (degree n).
  - Determine roots to $det(A-\lambda I)=0$, roots are eigenvalues λ.
  - Solve $(A-\lambda I)\,\mathbf{x}=0$ for each λ to obtain eigenvectors **x.**

Algorithm

- Finding the axes that minimize the projection errors and maximize the variance after the projection

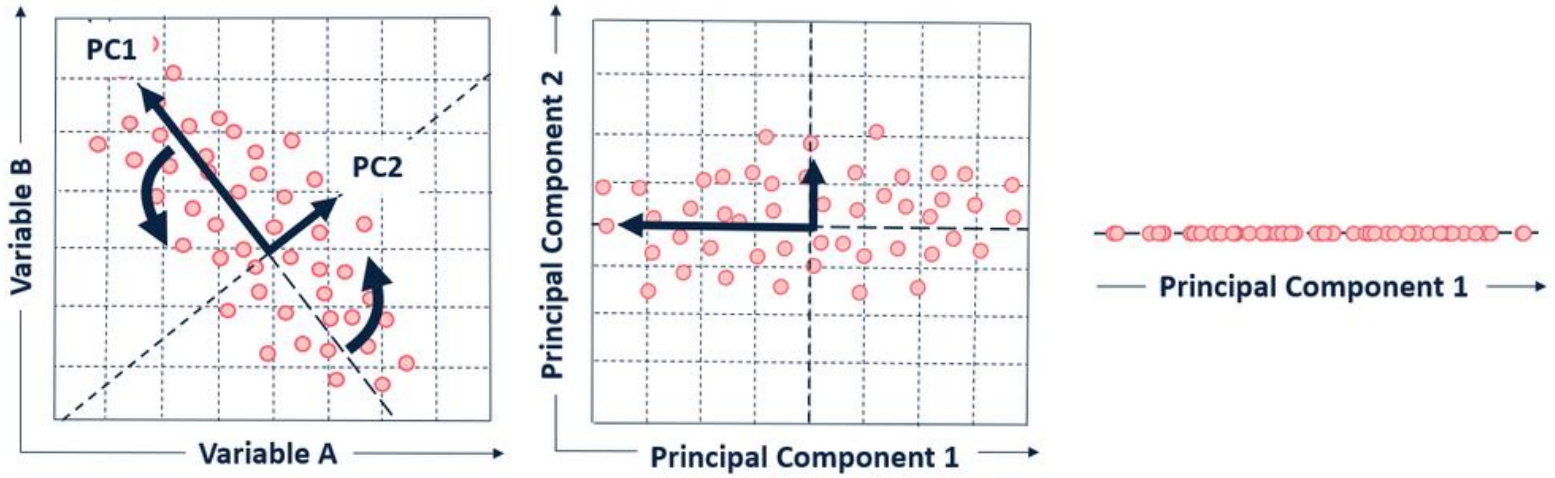


Fig. 5. Visual representation of principal components one and two that fit on the dataset. https://tinyurl.com/5czfny39 [10].

PCA with Python

- The purpose of principal component analysis is to find the best low-dimensional representation of the variation in a multivariate data set. PCA not only removes some redundancy, but also improves the variance of the data set.

  ✓PCA not only removed some redundancy but also improved variance in the dataset.

- For example, in the case of the **wine** data set, we have 13 chemical concentrations describing wine samples from three different cultivars.

- **"wine.csv"** [12] dataset has a large number of features (columns), PCA can help in reducing the dimensionality by transforming the data into a set of linearly uncorrelated variables (principal components). This is particularly useful when dealing with datasets where the number of features is significantly greater than the number of observations.

PCA with Python (Considering wine.csv dataset)

(https://tinyurl.com/xa8f35u6 [13])

- Considering **wine.csv** dataset
- **Step 1:** Import the libraries.

```
import numpy as nmp
import matplotlib.pyplot as mpltl
import pandas as pnd
```

- **Step 2:** Import the dataset (wine.csv).

```
DS = pnd.read_csv('Wine.csv')
# Now, distribute the dataset into two components "X" and "Y"
X = DS.iloc[: , 0:13].values
Y = DS.iloc[: , 13].values
```

	Wine	Alcohol	Malic.acid	Ash	Acl	Mg	Phenols	Flavanoids	Nonflavanoid.phenols	Proanth	Color.int	Hue	OD	Proline
2	1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
3	1	13.2	1.78	2.14	11.2	100	2.65	2.76	.26	1.28	4.38	1.05	3.4	1050
4	1	13.16	2.36	2.67	18.6	101	2.8	3.24	.3	2.81	5.68	1.03	3.17	1185
5	1	14.37	1.95	2.5	16.8	113	3.85	3.49	.24	2.18	7.8	.86	3.45	1480
6	1	13.24	2.59	2.87	21	118	2.8	2.69	.39	1.82	4.32	1.04	2.93	735
7	1	14.2	1.76	2.45	15.2	112	3.27	3.39	.34	1.97	6.75	1.05	2.85	1450
8	1	14.39	1.87	2.45	14.6	96	2.5	2.52	.3	1.98	5.25	1.02	3.58	1290
9	1	14.06	2.15	2.61	17.6	121	2.6	2.51	.31	1.25	5.05	1.06	3.58	1295
10	1	14.83	1.64	2.17	14	97	2.8	2.98	.29	1.98	5.2	1.08	2.85	1045
11	1	13.86	1.35	2.27	16	98	2.98	3.15	.22	1.85	7.22	1.01	3.55	1045
12	1	14.1	2.16	2.3	18	105	2.95	3.32	.22	2.38	5.75	1.25	3.17	1510

Fig. 6. wine.csv https://tinyurl.com/bdfakpab [12]

- **Step 3:** Split the dataset into the training set and testing set.

```
from sklearn.model_selection
import train_test_split as tts
X_train, X_test, Y_train, Y_test = tts(X, Y, test_size = 0.2, random_state = 0)
```

- **Step 4:** Feature Scaling (reprocessing on the training and test set (for example, standard scale fitting.)

```
from sklearn.preprocessing import StandardScaler as SS
SC = SS()
X_train = SC.fit_transform(X_train)
X_test = SC.transform(X_test)
```

PCA with Python (cont'd)

Step 5: Then, Apply the PCA function (Apply the PCA function into the training set and testing set for analysis.)

```
from sklearn.decomposition import PCA
PCa = PCA (n_components = 1)
X_train = PCa.fit_transform(X_train)
X_test = PCa.transform(X_test)
explained_variance = PCa.explained_variance_ratio_
```

Step 6: Fit Logistic Regression for the training set.

```
from sklearn.linear_model import LogisticRegression as LR
classifier_1 = LR (random_state = 0)
classifier_1.fit(X_train, Y_train)
LogisticRegression(random_state=0)
```

Step 7: Predict the testing set result.

```
Y_pred = classifier_1.predict(X_test)
```

Step 8: create the confusion matrix.

```
from sklearn.metrics import confusion_matrix as CM
c_m = CM (Y_test, Y_pred)
```

PCA with Python (cont'd)

- **Step 9:** Predict the result of the training set.

```
from matplotlib.colors import ListedColormap as LCM
X_set, Y_set = X_train, Y_train
X_1, X_2 = nmp.meshgrid(nmp.arange(start = X_set[:, 0].min() - 1,
stop = X_set[: , 0].max() + 1, step = 0.01),
nmp.arange(start = X_set[: , 1].min() - 1,
stop = X_set[: , 1].max() + 1, step = 0.01))
```

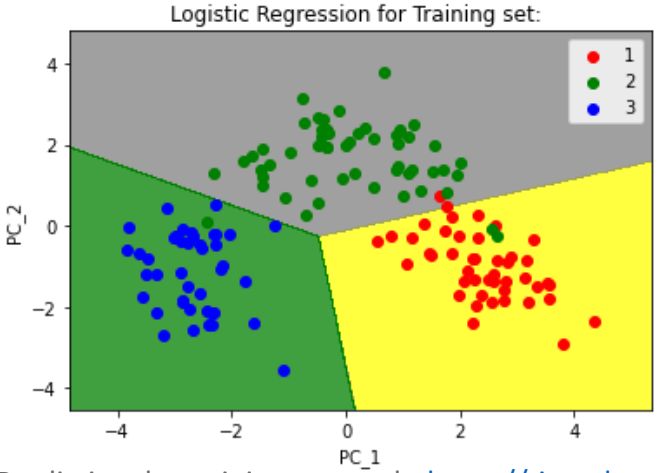


Fig. 7. Predicting the training set result. https://tinyurl.com/xa8f35u6 [13]

- **Step 10:** Visualize the result of the testing set.

```
from matplotlib.colors import ListedColormap as LCM
X_set, Y_set = X_test, Y_test
X_1, X_2 = nmp.meshgrid(nmp.arange(start = X_set[: , 0].min() - 1,
stop = X_set[: , 0].max() + 1, step = 0.01),
nmp.arange(start = X_set[: , 1].min() - 1,
stop = X_set[: , 1].max() + 1, step = 0.01))
mpltl.contourf(X_1, X_2, classifier_1.predict(nmp.array([X_1.ravel(),
X_2.ravel()]).T).reshape(X_1.shape), alpha = 0.75,
```

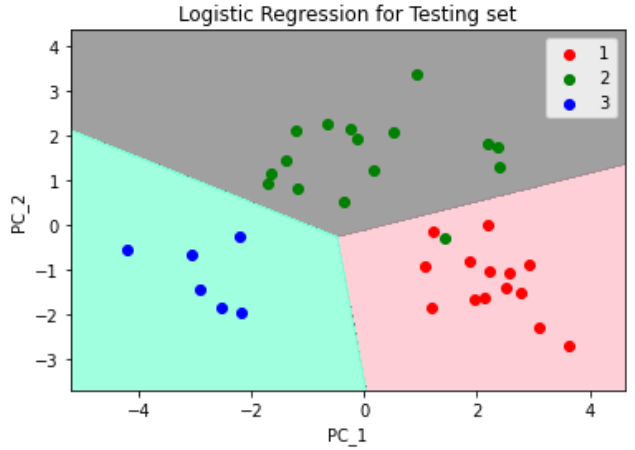


Fig. 8. Visualizing the Test set results. https://tinyurl.com/xa8f35u6 [13]

University of Windsor

Applications

- Example applications:(https://tinyurl.com/4nyk94p9 [14])
  - **Computer Vision**
    - Representation
    - Pattern Identification
    - Image compression
    - Face recognition
  - **Gene expression analysis**
    - ***Purpose:*** Determine core set of conditions for useful gene comparison
  - **Handwritten character recognition**
  - **Data Compression, etc.**

Conclusion

- PCA can be useful when there is a severe high-degree of correlation present in the multi-attributes.
- When a data set consists of several *clusters*, the principal axes found by PCA usually pick projections with good separation. PCA provides an effective basis for *feature extraction* in this case.
- For good data compression, PCA offers a useful self-organize learning procedure
- PCA requires to diagonalise matrix $\boldsymbol{C}$ (dimension:n x n). Heavy if n is large !
- PCA only finds linear sub-spaces
- It works best if the individual components are Gaussian-distributed
- PCA does not say how many target dimensions to use

References

1. http://wiki.pathmind.com/eigenvector
2. https://www.geeksforgeeks.org/ml-principal-component-analysispca/
3. https://blog.clairvoyantsoft.com/eigen-decomposition-and-pca-c50f4ca15501
4. https://www.turing.com/kb/guide-to-principal-component-analysis
5. https://en.wikipedia.org/wiki/Dimensionality_reduction
6. https://www.geeksforgeeks.org/dimensionality-reduction/
7. https://www.pinecone.io/learn/dimensionality-reduction/
8. https://towardsdatascience.com/pca-clearly-explained-how-when-why-to-use-it-and-feature-importance-a-guide-in-python-7c274582c37e
9. https://erdogant.github.io/pca/pages/html/Algorithm.html
10. https://community.alteryx.com/t5/Data-Science/Tidying-up-with-PCA-An-Introduction-to-_Principal-Components/ba-p/382557
11. https://bookdown.org/andreabellavia/mixtures/principal-component-analysis.html
12. https://gist.github.com/tijptjik/9408623#file-wine-csv
13. https://www.geeksforgeeks.org/principal-component-analysis-with-python/
14. https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.09-Principal-Component-Analysis.ipynb

Thank you for attention