PCA

Machine Learning

SAIF ALI JAN
24065142

Principal Component Analysis (PCA): How Many Components Should You Keep?

PCA is an original, unsupervised method of dimensionality-reduction, a fundamental procedure in which one tries to maintain maximum information in the data set and minimizes the dimensionality at the same time. In essence, PCA is a transformation of the initial coordinate system in which orthogonal vectors, which are called principal components, have their variance maximised. These are based on the eigenvectors of the covariance matrix of data and the values of the eigenvalues tell how much variance is contained in a particular component (Sarkar, 2023). One particular issue with the application of components in PCA is the question of how many results to keep, to ensure that critical information is captured, but not too many as this may remove potentially important data, and it is this that directly relates to both visualisation effectiveness and the effect on the resultant learning since too few components can lead to the loss of important information, and too many leads to unnecessary noise and worse interpretation.

One of the major retention criteria is explained variance. Explained variance This value represents the percentage of the total variance available in the original data that each of the principal components can explain. Arranging elements in decreasing order of explained variance will provide a series with the first one emphasizing the highest variability, the following one the next variability, and so on. The cumulative explained variance is normally calculated by researchers by summing up the explained variance ratios of a specified number of components (Bharadiya, 2023). The most frequently used heuristic is to use the fewest number of components that together have a large percentage of total variance 90 or 95 per cent of the information in an acceptable loss is tolerated. Such cumulative method will guarantee dimensionality reduction will leave behind most of the informative variability within data.

A scree plot is a popular graphic diagnostic used to determine individual and cumulative explained variance. The x-axis in this plot represents the indices of the components (e.g, PC1, PC2, PC3 etc.), whereas the y-axis represents eigenvalues or proportions of explained variance. The contribution to variance can be expected to decrease with the index of components, and a plot may tend to flatten off with an elbow where new components are of marginal value. The elbow rule states that the components before this bend are of significant contribution to the explanatory power with the addition of the components after the elbow being marginal in the provision of information. As a result, numerous practitioners would take this aspect slowly and leave the entire part until--

but not beyond--the elbow to avoid complicate the statistical faithfulness (Hasan and Abdulazeez, 2021).

A quantitatively based approach is to have a target towards a cumulative accumulation of explained variance, e.g. require that retained components collectively explain at least 90 percent of total variance. This threshold offers a objective method of selection of components that is not based on a mere visual evaluation. A lot of PCA systems permit explicit specification of a desired variance threshold to use in model fitting which is then automatically determined by the algorithm. The consistent methods are the Kaiser rule that encourages retaining terms whose eigenvalue is greater than one and assumes that they explain more variance than one single original variable. The conceptual simplicity of the Kaiser rule is mostly limited to standardized datasets although in certain cases it can lead to over- or under-selection of components (Greenacre et al., 2022).

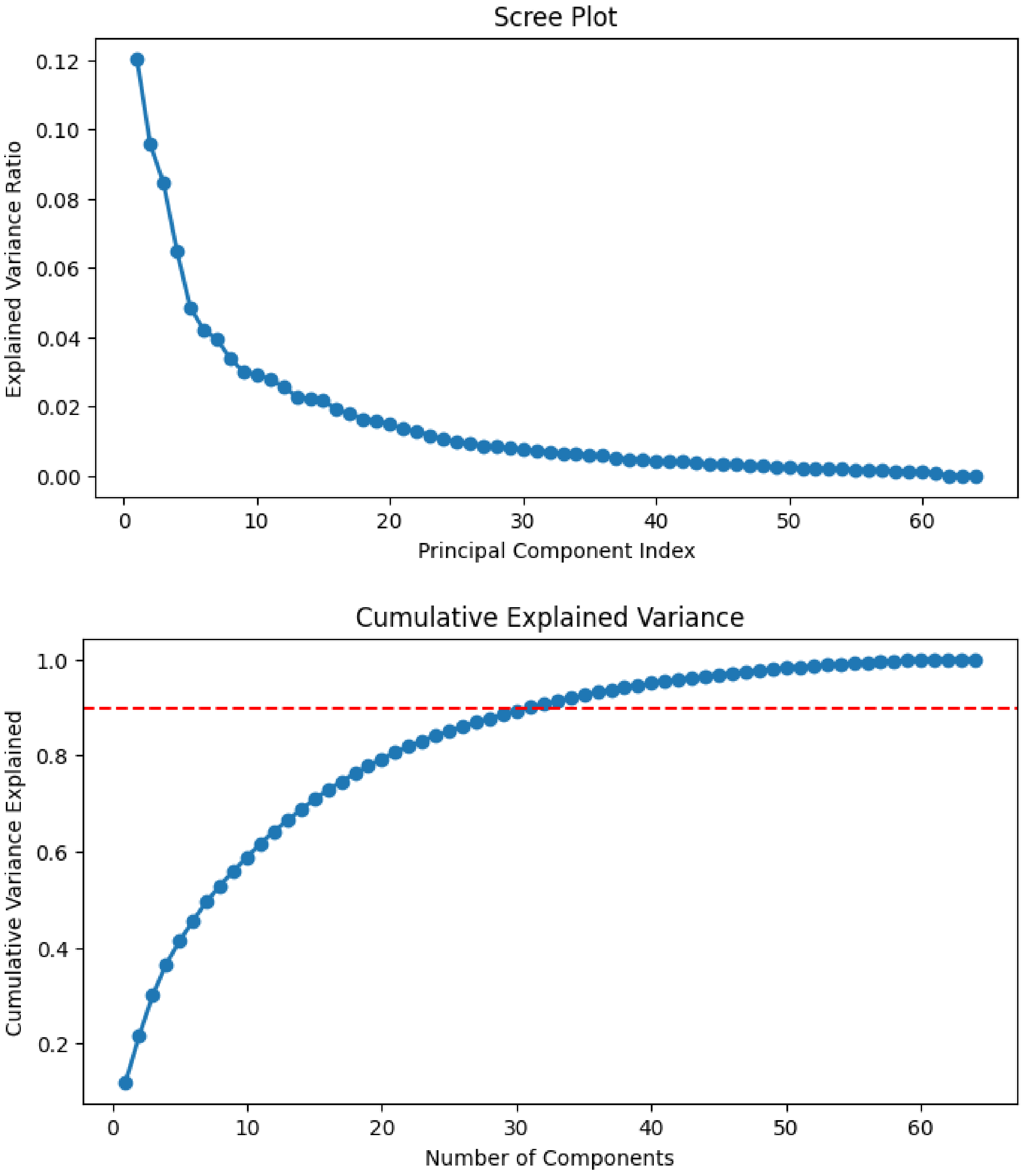
Despite the utilization of explained variance and scree plot based methods, there is more insight into how best the component selection can work, this is achieved through the analysis of an error on reconstruction. PCA can be viewed as a lossy compression scheme: The data are projected onto the sub space covered by the first k components and later it is reconstructed again into the original high dimensional space. The fidelity to this reduced representation is represented by the reconstruction error, frequently measured in terms of the sum of squared deviations or root-mean-square error. A smaller error in reconstruction will indicate that the selected components will represent the inherent structure of the original data sufficiently. Reconstruction error can give a concrete measure on how well reduced representations represent the original in such applications as image compression, where reconstructed images created using different numbers of principal components could be visually inspected to explain how much visual information was retained (Bharadiya, 2023).

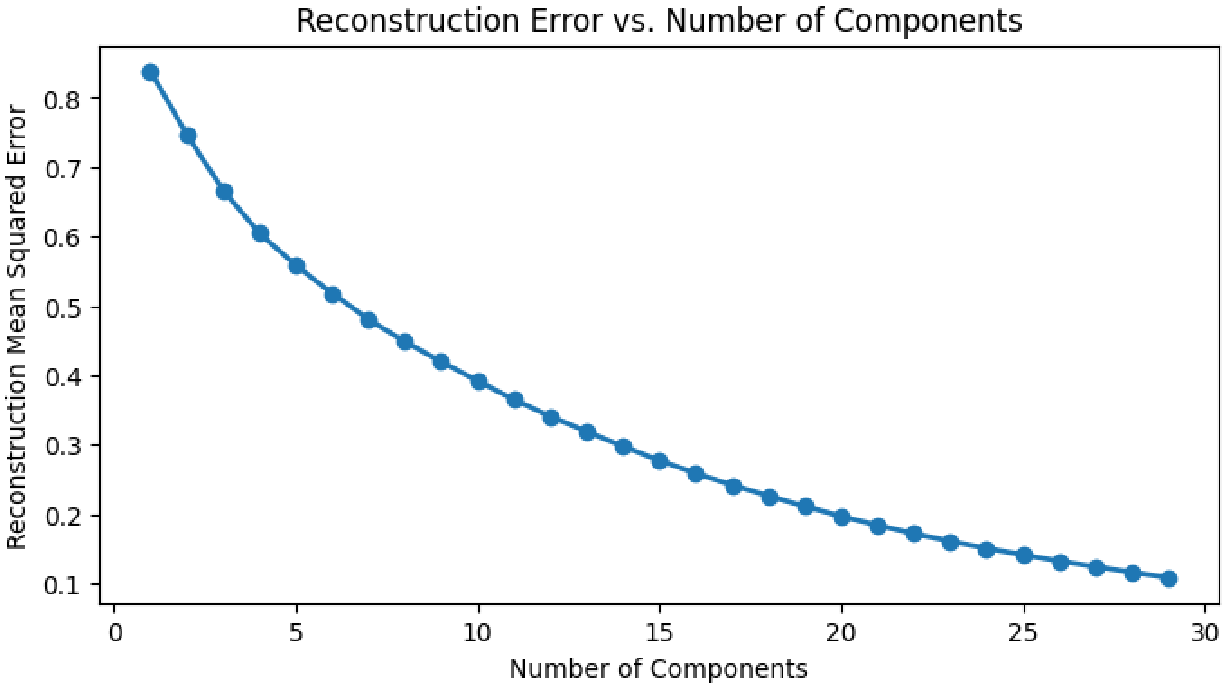
In a machine-leaning pipeline, the process of determining the right amount of principal components is usually coded into the model-evaluation loop. Instead of using explained-variance measures or scree plots for evaluation, one can see how changes in the number of components used as input affect the results of downstream tasks, e.g. whether a classifier trained using PCA-reduced features still continues to improve itself with the addition of more components. This kind of task-based comparing is especially beneficial in cases where the final goal of PCA is the improvement of a certain learning criterion and not dimensionality reduction itself.

To realise these considerations, the script below provides an illustration of using PCA on a tabular data, including the steps of producing scree diagrams, cumulative-explained-variance plots, and the calculation of reconstruction error. The example is based on the canonical digits dataset that is provided in scikit-learn, but the process is easily generalizable to other high-dimensional imaging datasets like facial images:

Github: https://github.com/saifalijan/Machine-Learning-Individual

```
# PCA Component Selection: Python Demo with Scree Plot, Explained
Variance, and Reconstruction

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

# Load tabular data
digits = load_digits()
X = digits.data  # shape (n_samples, n_features)

# Standardize data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Fit PCA without reducing dimensions to get all principal components
pca = PCA()
X_pca = pca.fit_transform(X_scaled)
explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)

# Scree plot (variance explained per component)
plt.figure(figsize=(8, 4))
plt.plot(range(1, len(explained_variance)+1), explained_variance, 'o-',
linewidth=2)
plt.title("Scree Plot")
plt.xlabel("Principal Component Index")
plt.ylabel("Explained Variance Ratio")
plt.show()

# Cumulative variance plot
plt.figure(figsize=(8, 4))
plt.plot(range(1, len(cumulative_variance)+1), cumulative_variance, 'o-',
linewidth=2)
plt.title("Cumulative Explained Variance")
plt.xlabel("Number of Components")
plt.ylabel("Cumulative Variance Explained")
plt.axhline(0.90, color='red', linestyle='--')  # 90% line
plt.show()

# Reconstruction error for different numbers of components
reconstruction_errors = []
component_range = range(1, 30)

for k in component_range:
    pca_k = PCA(n_components=k)
    X_proj = pca_k.fit_transform(X_scaled)
    X_reconstructed = pca_k.inverse_transform(X_proj)
    error = np.mean((X_scaled - X_reconstructed) ** 2)
    reconstruction_errors.append(error)

plt.figure(figsize=(8, 4))
plt.plot(component_range, reconstruction_errors, 'o-', linewidth=2)
plt.title("Reconstruction Error vs. Number of Components")
plt.xlabel("Number of Components")
plt.ylabel("Reconstruction Mean Squared Error")
plt.show()
```


Scree Plot
Explained Variance Ratio
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
50
60
Principal Component Index
Cumulative Explained Variance
Cumulative Variance Explained
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50
60
Number of Components



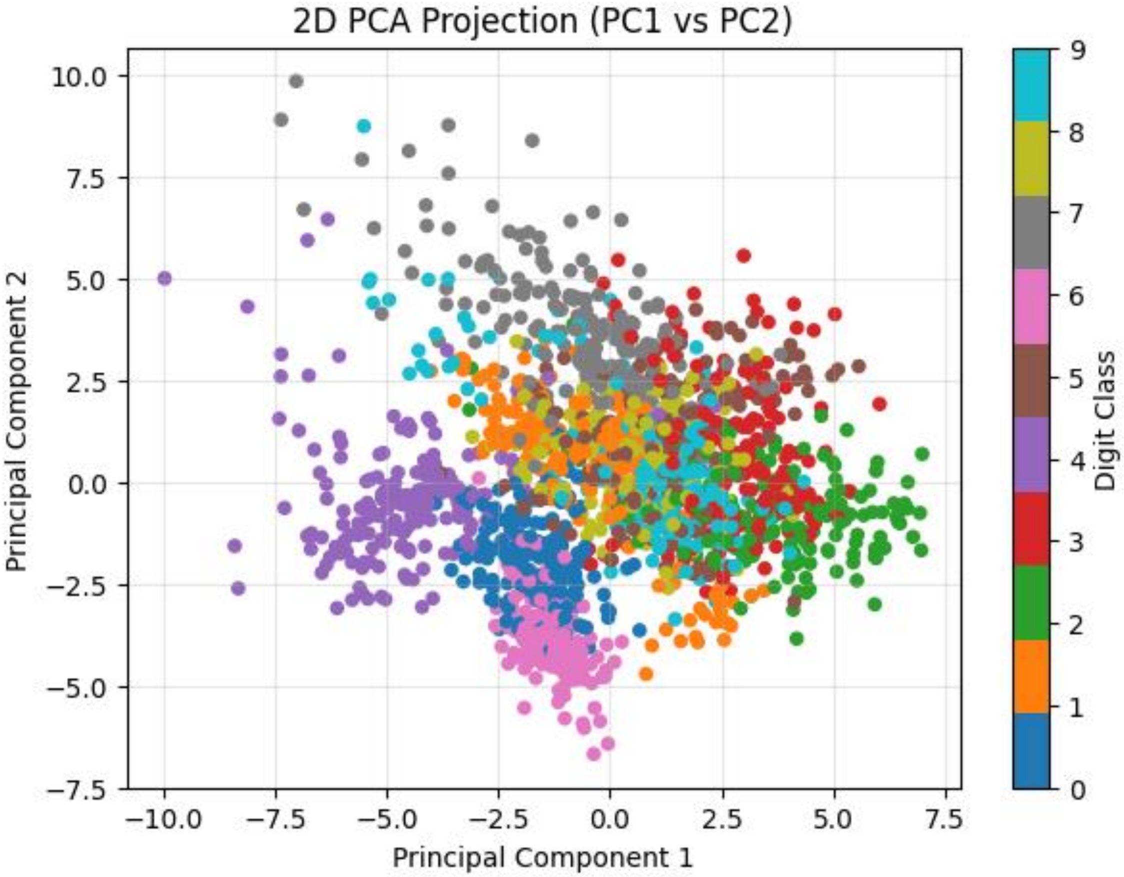
First, the code normalises the vector of features as the unsanitised variables may bias the PCA subspace, since the variables do not have homogenous scales. After that, the whole data are estimated to the PCA estimator producing a scree plot as well as cumulative-variance plot. The reconstruction error analysis further shows how data approximation gradually becomes better with the size of retained dimensions becoming increasingly large. Researchers can use such visualisations in determining an appropriate dimensionality that balances the reduction and information loss (Sarkar, 2023).

In short, even though PCA is an efficient dimensionality-reduction approach, the best choice of the number of principal-components requires the use of evidence-based decision-making. The scree chart and the elbow rule provide an intuitive, graphic, heuristic of identifying disappearing returns, but explicit quantitative values on the eliminated variance can be used to provide an even tighter criterion. These auxiliary measures like reconstruction error and task dependent performance also provide supplemental information to the trade-off between dimensionality and fidelity hence giving practitioners a conceptual frame of reference to incorporate PCA into their pipeline of analysis.

The Implications of Component Selection in the Real World of machine learning.

The choice of the right amount of principal components is not only an academic issue, but it directly impacts the actual machine-learning processes, particularly in cases of high-dimensional data. The datasets are usually hundreds or even thousands of features in a wide variety of applied contexts, including image classification, sensor monitoring, genomic analysis, and financial modelling. Such large feature spaces may be computationally expensive to process, and may have a negative impact on model performance in the case of irrelevant or redundant patterns. PCA is useful in optimising such workflows and reduces the original data to a smaller number of orthogonal components that best represent the most salient data and excludes noise.

Embarking on a dimensionality reduction step that is selected well can greatly lower the model training time, especially when a feature dimensionality-sensitive algorithm, e.g. Support Vector Machines, K-Nearest Neighbours, and Gradient Boosting, is targeted. Reducing the feature space to a small set of major components allows it to use less memory and speed up and stabilize training pipelines. Furthermore, PCA may be used as a preprocessing process which can alleviate the problem of multicollinearity which is common in regression based models with correlated variables which result in unstable parameter estimates. PCA allows algorithms to be run on a well-conditioned set of predictors by means of decorrelation of the feature space. Even though PCA is frequently applied to numerical compression only, the idea of its usefulness to depict latent structure (i.e., finding correlations, segments or latent patterns) can also be considered vital. This process of choosing the number of components is thus associated with not only optimisation of downstream modelling but also the quality of information obtained on the data. In most real-world applications, it can be useful to choose fewer components and use them to visualise the data, e.g. two or three-dimensional PCA embeddings to explore the distribution of data.



Restrictions, limitations and Ethical Implications.

PCA notwithstanding its extensive usage has limitations. Its linearity assumption is one of the most crucial ones: PCA is based on linear transformations and thus cannot be applied to datasets the structure of which is determined by nonlinear relations. Nonlinear methods, such as Kernel PCA, t-SNE, or UMAP, can be more effective in such situations and would reflect the structure of the manifolds. As a result, PCA can be very robust and interpretable, however, it can be weak in other areas like speech processing, some types of image datasets or very complicated hierarchical relationships in the dataset.

The other difficulty is a scale-sensitivity of the PCA. Normalisation or standardisation of the data must be carried out prior to the execution of PCA since variables with big ranges will overwhelm the principal components. Loss of this causes the misrepresentation of estimates of variance and biases in the process of component selection. Though your tutorial is properly standardised, practitioners who are not familiar with this criterion may accidentally come up with erroneous conclusions of their PCA results.

There is also a disadvantage of interpretability. Although PCA generates mathematically clean manifestations of variance, the transformed elements are frequently abstract and do not have a direct meaning. Principal components are weighted combinations unlike the original features which are explicit measurements whose meaning can be hard to describe succinctly.

Making PCA a Full Machine-Learning Pipeline.

The quality of the end-model will be based not only on the number of components used but also on the interaction between PCA and the other processes of the pipeline. In general, PCA is to be added to the data after data cleaning and scaling but before fitting the model. Tuned as a hyper-parameter when used together with other techniques such as cross-validation, PCA can be used by practitioners to determine how varying numbers of components impact model accuracy, stability, and generalisation.

To give an example, a classifier can give 15 principal components that are optimal, although 20 principal components explain a greater proportion of variance. This is an indication that the explained variance is not a determinant of an optimal predictive performance. Rather, the desired number of components should be in line with model objectives, computation requirements and accuracy. More than that, PCA becomes an inherent aspect of regularisation strategies. This is the reason why PCA can be of great use in situations with a small dataset where overfitting may become an issue. In these cases, few components can be chosen to increase generalisation and minimise variance error in predictive tasks. Lastly, PCA can be easily combined with deployment and monitoring phases. The PCA transformation may be stored in preprocessing pipeline after having a model operationalized to make sure that incoming data are projected to the same component space. This stabilises the long term model behaviour, aids its monitoring, and ensures consistency between the training and production environment.

References

Bharadiya, J.P., 2023. A tutorial on principal component analysis for dimensionality reduction in machine learning. *International journal of innovative science and research technology*, *8*(5), pp.2028-2032.

Greenacre, M., Groenen, P.J., Hastie, T., d'Enza, A.I., Markos, A. and Tuzhilina, E., 2022. Principal component analysis. *Nature Reviews Methods Primers*, *2*(1), p.100.

Hasan, B.M.S. and Abdulazeez, A.M., 2021. A review of principal component analysis algorithm for dimensionality reduction. *Journal of Soft Computing and Data Mining*, *2*(1), pp.20-30.

Sarkar, S.K., 2023. Principal component analysis. *Statistical Procedures for Analyzing Agricultural Data using R*, p.139.

Jolliffe, I.T. and Cadima, J., 2016. Principal component analysis: a review and recent developments. Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences, 374(2065), p.20150202.

Ashraf, M., Anowar, F., Setu, J.H., Chowdhury, A.I., Ahmed, E., Islam, A. and Al-Mamun, A., 2023. A survey on dimensionality reduction techniques for time-series data. *IEEE Access*, *11*, pp.42909-42923.