Advanced Data Analysis

DATA 71200

Class 13: Unsupervised Learning (Principal Component Analysis, Non-Negative Matrix Factorization, and Manifold Learning)

Unsupervised Learning

- **A set of algorithms that learn representations or partitions of unlabeled data**
  - In the absence of labels, evaluation is challenging
    - Often performed through visualization
- **Useful for**
  - Exploratory data analysis
  - Pre-processing data

Curse of Dimensionality

- **Large number of features makes training slow and it is hard to find robust solutions**
- **High-dimensional representations tend to be sparser than low-dimensional representations**
  - Meaning that new data tends to be further away than existing data — increasing the risk of overfitting

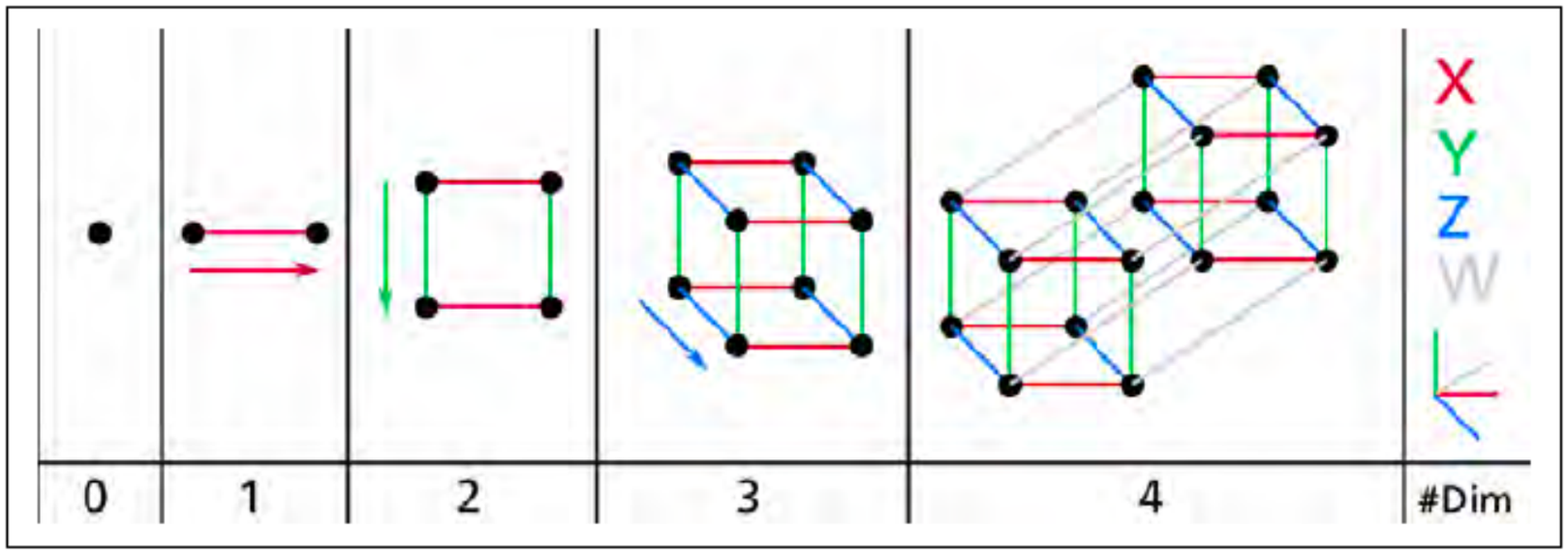

Figure 8-1. Point, segment, square, cube, and tesseract (0D to 4D hypercubes)[2]

Géron, Aurélien. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly Media, Inc.

Principal Component Analysis (PCA)

- **PCA can be used for *projection***
  - *Projection* takes advantage of the fact that most data is not uniformly distributed across the dimensions
    - Rather correlations exist
- **PCA exploits these correlations by finding directions that capture the maximum amount of variance in the data**
    - These directions, or principle components, are orthogonal eigenvectors

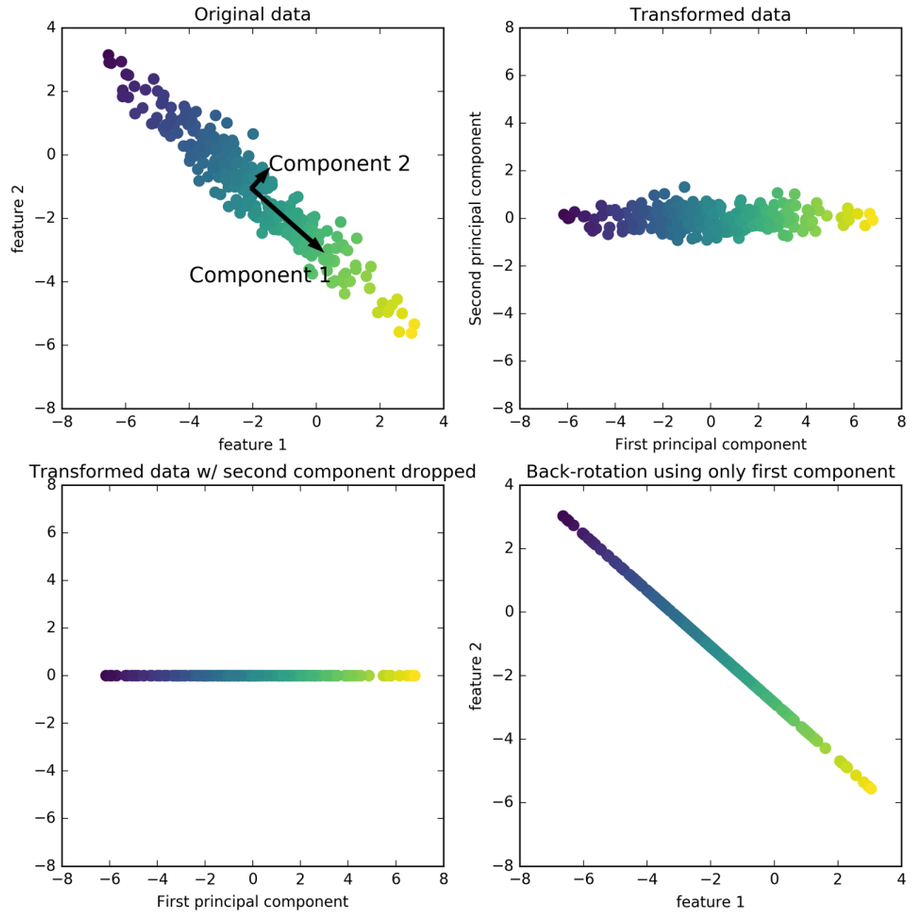

Figure 3-3. Transformation of data with PCA

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Principal Component Analysis (PCA)

- **PCA can be used for *visualization***

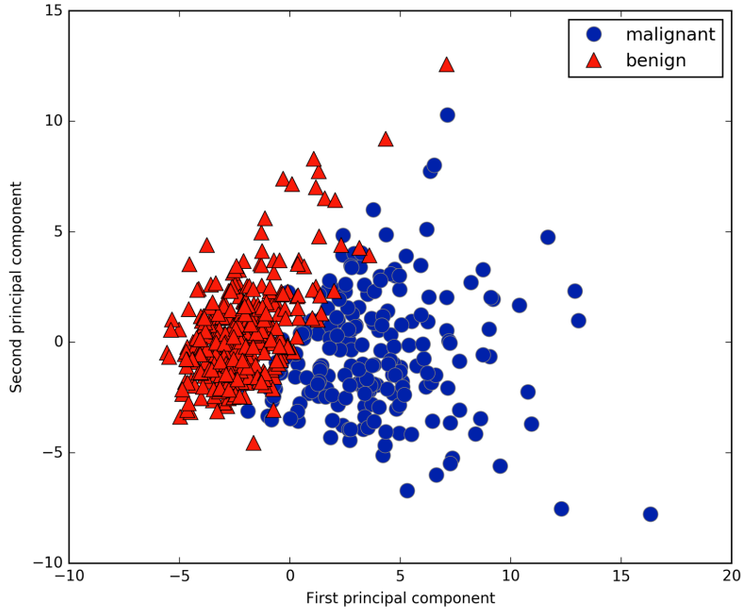

Figure 3-5. Two-dimensional scatter plot of the Breast Cancer dataset using the first two principal components

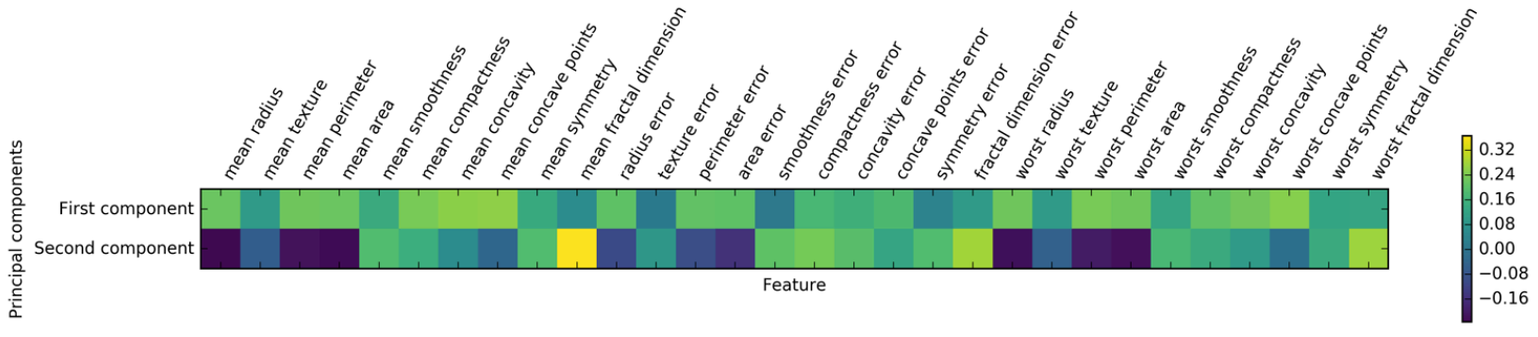

Figure 3-6. Heat map of the first two principal components on the Breast Cancer dataset

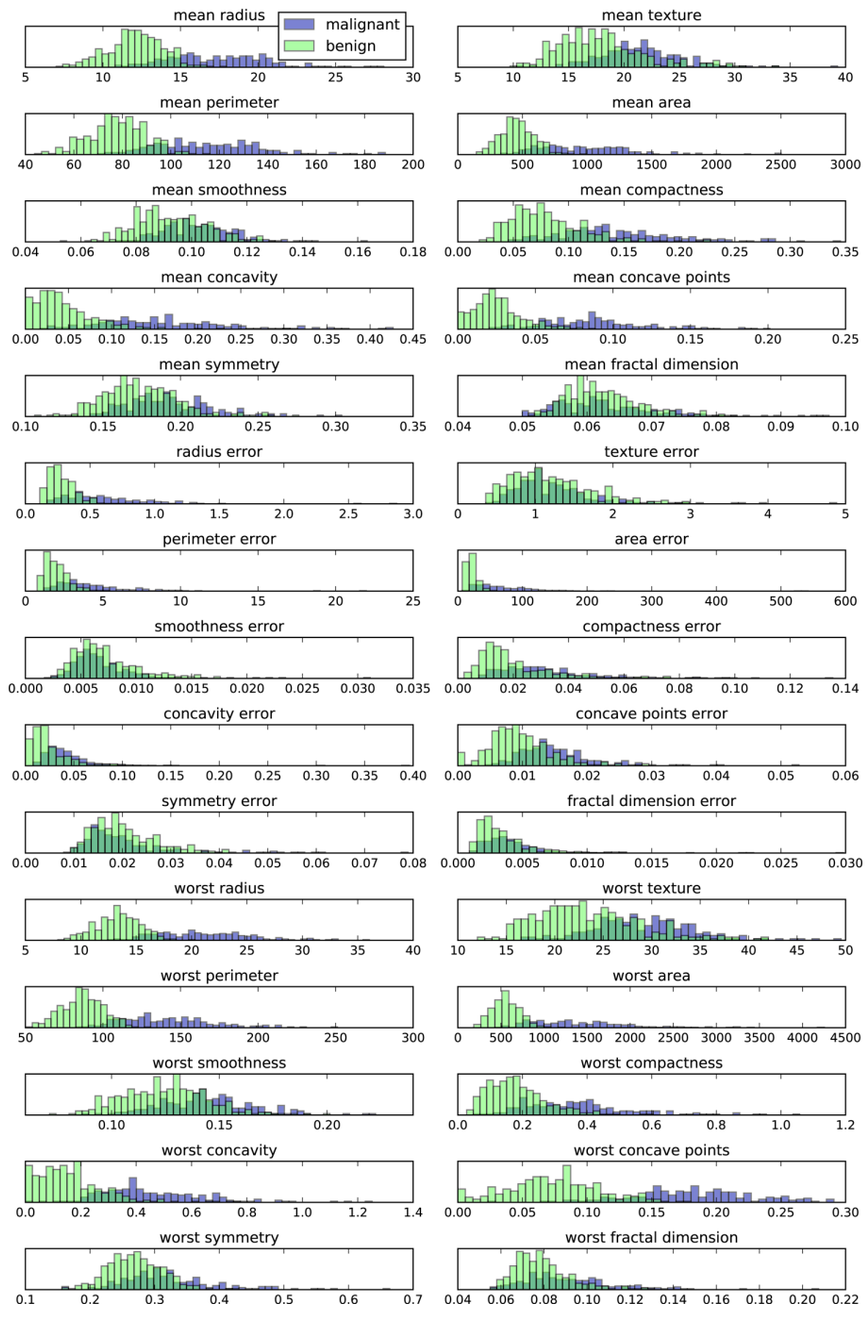

Figure 3-4. Per-class feature histograms on the Breast Cancer dataset

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Principal Component Analysis (PCA)

- **PCA can be used for *feature extraction***
- **Whitening transforms the data to the same scale**
  - also de-correlates the data

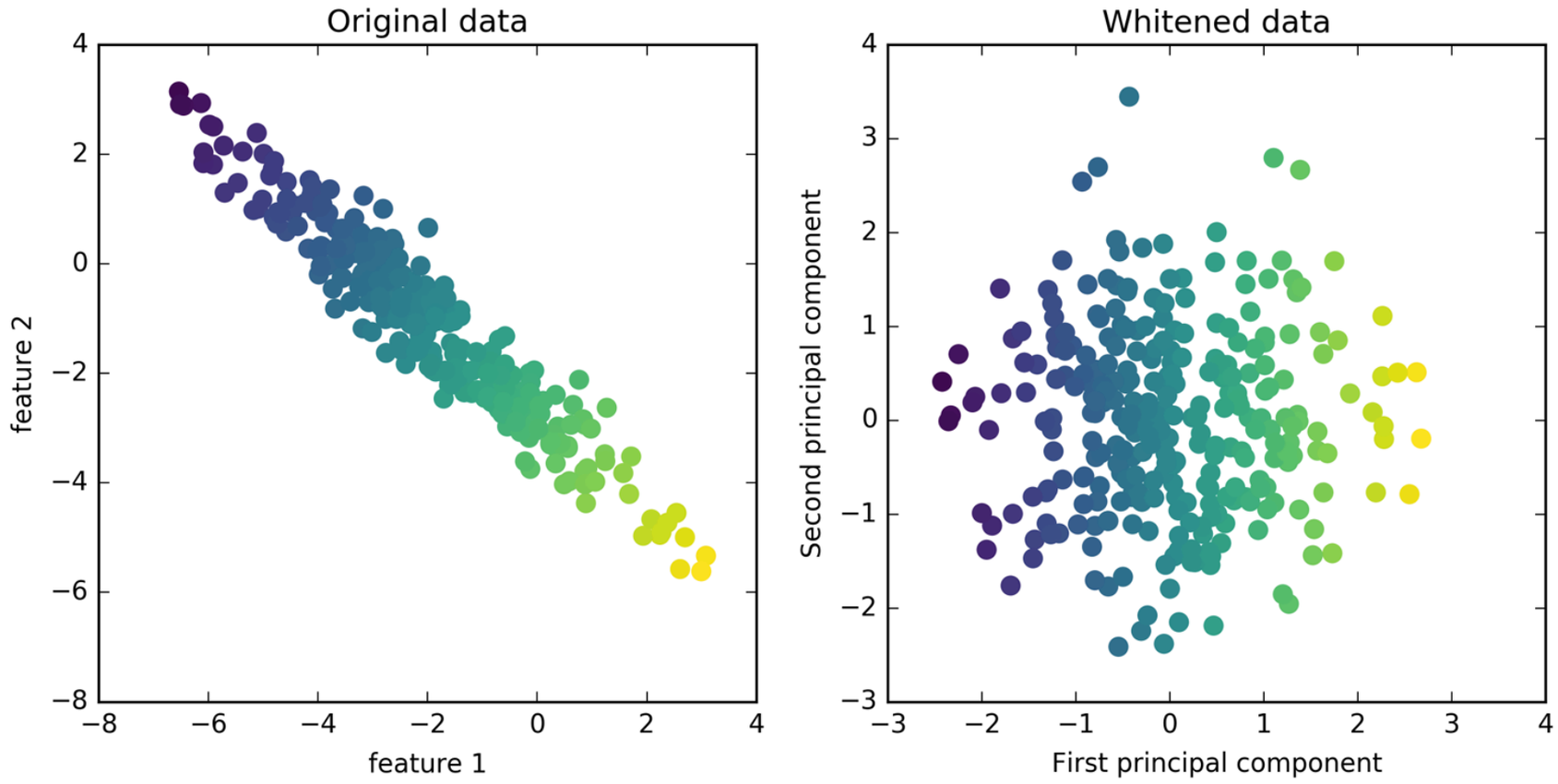

Figure 3-8. Transformation of data with PCA using whitening

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Principal Component Analysis (PCA)

Figure 3-7. Some images from the Labeled Faces in the Wild dataset

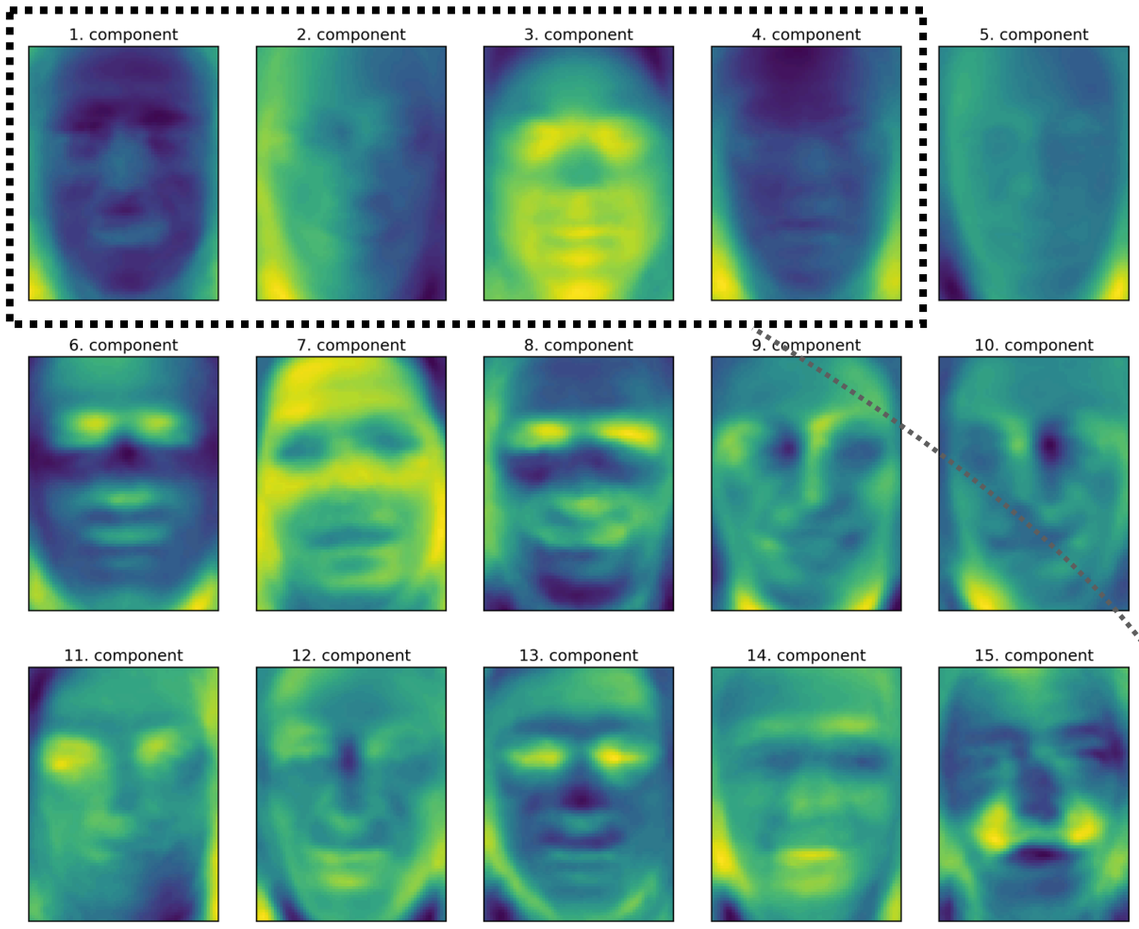

Figure 3-9. Component vectors of the first 15 principal components of the faces dataset

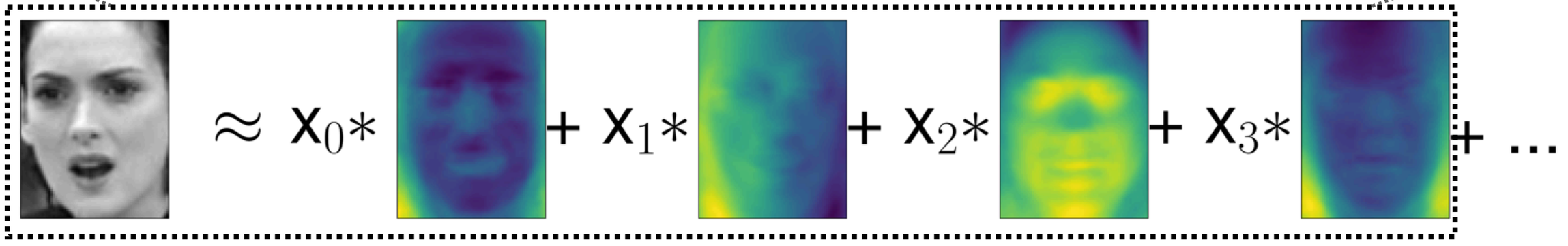

Figure 3-10. Schematic view of PCA as decomposing an image into a weighted sum of components

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Principal Component Analysis (PCA)

Figure 3-11. Reconstructing three face images using increasing numbers of principal components

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Principal Component Analysis (PCA)

- **Variants**
  - Incremental PCA
    - split the data into mini-batches and feed an IPCA algorithm one mini-batch at a time
  - Randomized PCA
    - quick approximation of the first *d* components
  - Kernel PCA
    - complex nonlinear projections for dimensionality reduction (same kernel trick as SVM)

Principal Component Analysis (PCA)

- **Best Practices**
  - Scale data to have unit variance
  - Dimensionality reduction: choose the number of dimensions that add up to a sufficiently large portion of the variance (e.g., 95%)
  - Visualization: limit to 2 or 3 dimensions
- **Strengths**
  - Removes correlations from data
  - Reduces overfitting
  - Speeds up processing time
- **Weaknesses**
  - Hard to interpret
  - Assumes linearity

Non-Negative Matrix Factorization (NMF)

- **Only works on zero or positively valued data**
  - Unlike PCA which can work on negative data as well
  - Makes the NMF bases more interpretable than PCA's
- **No ordering of the bases, as there is in PCA**

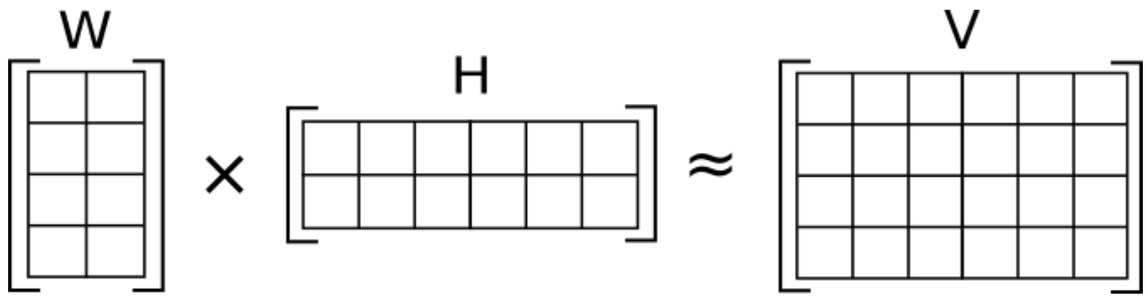

*Illustration of approximate non-negative matrix factorization: the matrix **V** is represented by the two smaller matrices **W** and **H**, which, when multiplied, approximately reconstruct **V***

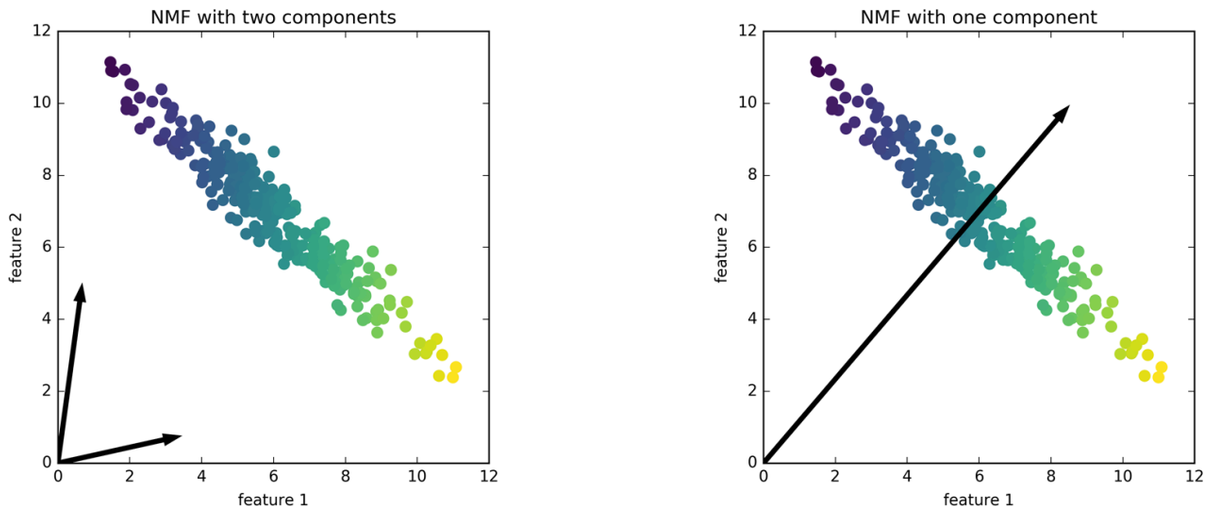

Figure 3-13. Components found by non-negative matrix factorization with two components (left) and one component (right)

https://en.wikipedia.org/wiki/Non-negative_matrix_factorization

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Non-Negative Matrix Factorization (NMF)

PCA

Figure 3-11. Reconstructing three face images using increasing numbers of principal components

NMF

Figure 3-14. Reconstructing three face images using increasing numbers of components found by NMF

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Non-Negative Matrix Factorization (NMF)

PCA

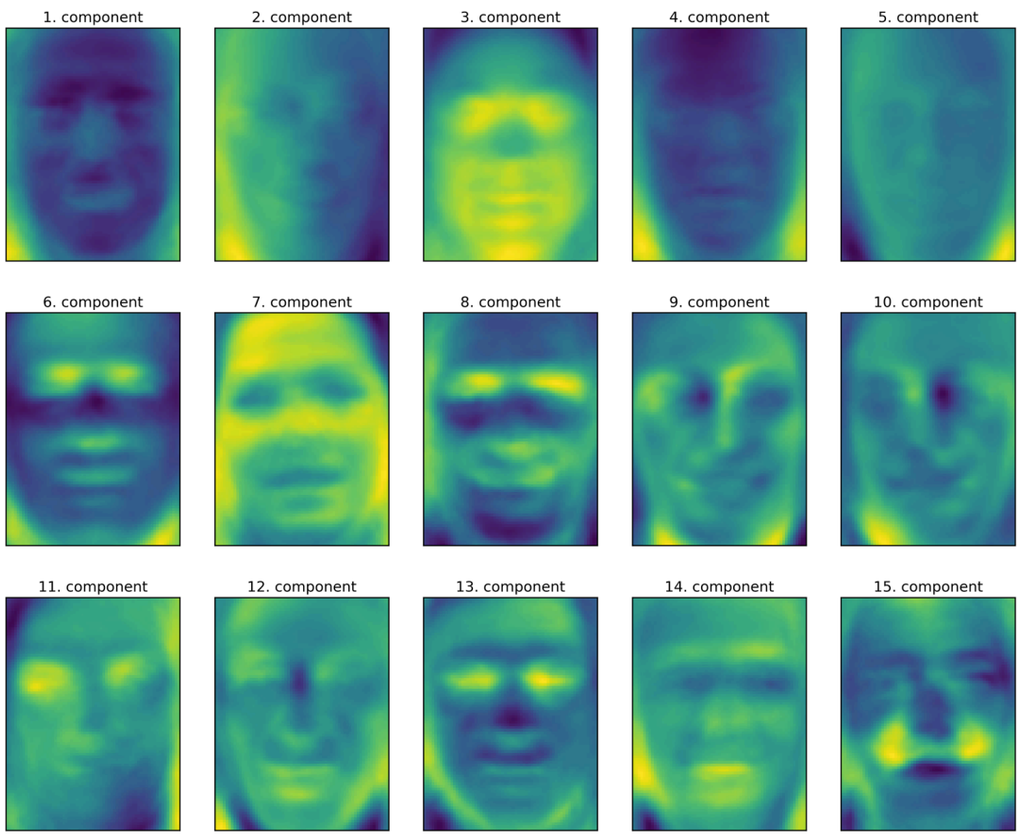

Figure 3-9. Component vectors of the first 15 principal components of the faces dataset

Figure 3-16. Faces that have a large coefficient for component 3

NMF

Figure 3-15. The components found by NMF on the faces dataset when using 15 components

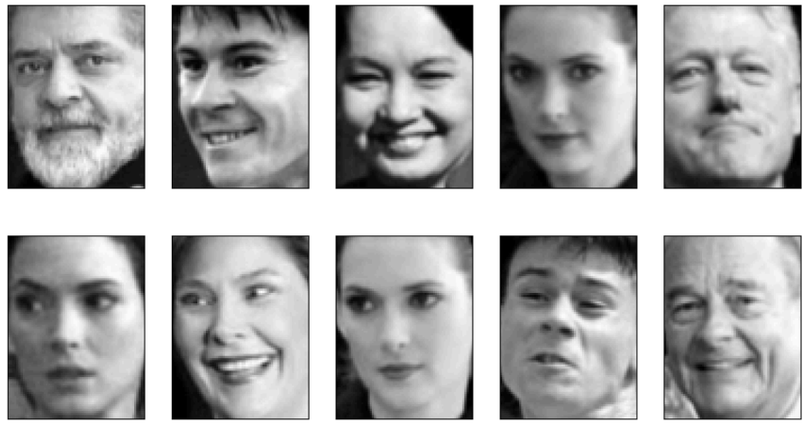

Figure 3-17. Faces that have a large coefficient for component 7

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Non-Negative Matrix Factorization (NMF)

- **NMF works better than PCA for decomposing data into its constituent parts**

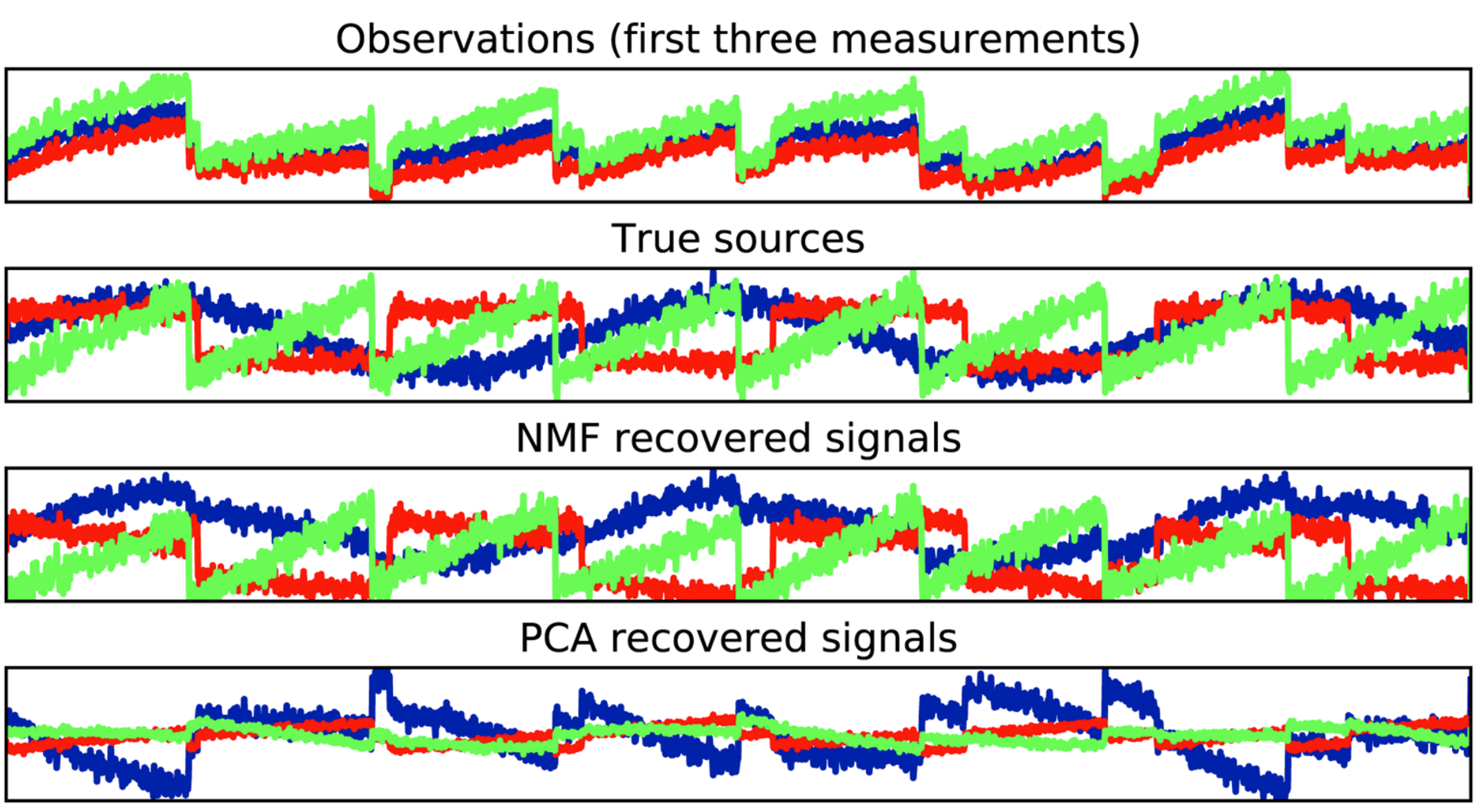

Figure 3-19. Recovering mixed sources using NMF and PCA

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Non-Negative Matrix Factorization (NMF)

- **Best Practices**
  - Specify initialization if you want reproducibility
- **Strengths**
  - Positive bases are more interpretable than negative
  - Decomposes data into its constituent parts
- **Weaknesses**
  - Bases values can vary based on starting point and number of dimensions
  - Doesn’t have a closed form solution

Manifold Learning

- **Manifold assumption - that most high-dimensional representations are close to a related low-dimensional representation**
- **t-SNE (t-Distributed Stochastic Neighbor Embedding) finds a 2D representation of the data that attempts to preserve distances between data points**
  - Particularly for data points that are close to one another

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Manifold Learning

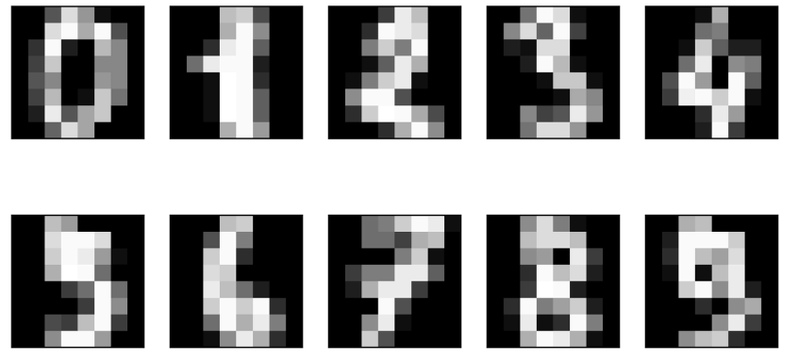

Figure 3-20. Example images from the digits dataset

PCA

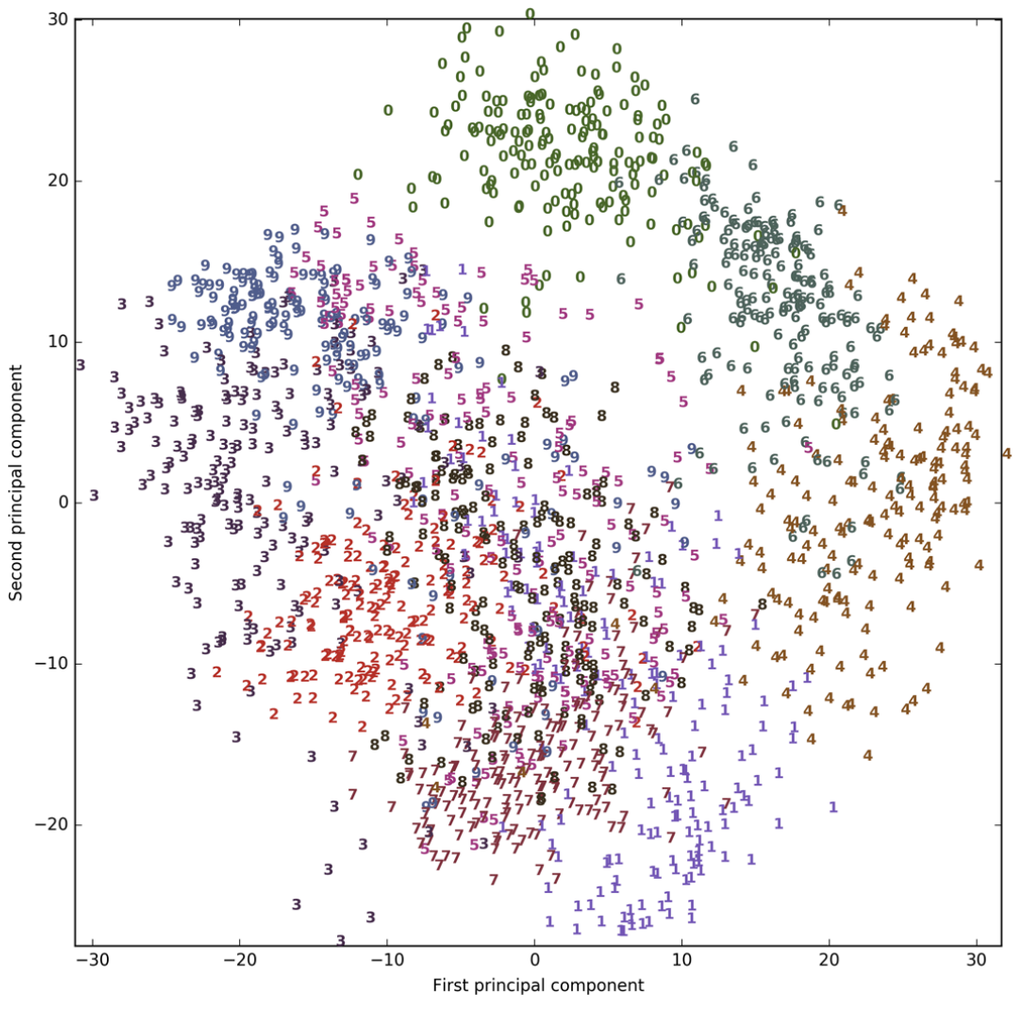

Figure 3-21. Scatter plot of the digits dataset using the first two principal components

t-SNE

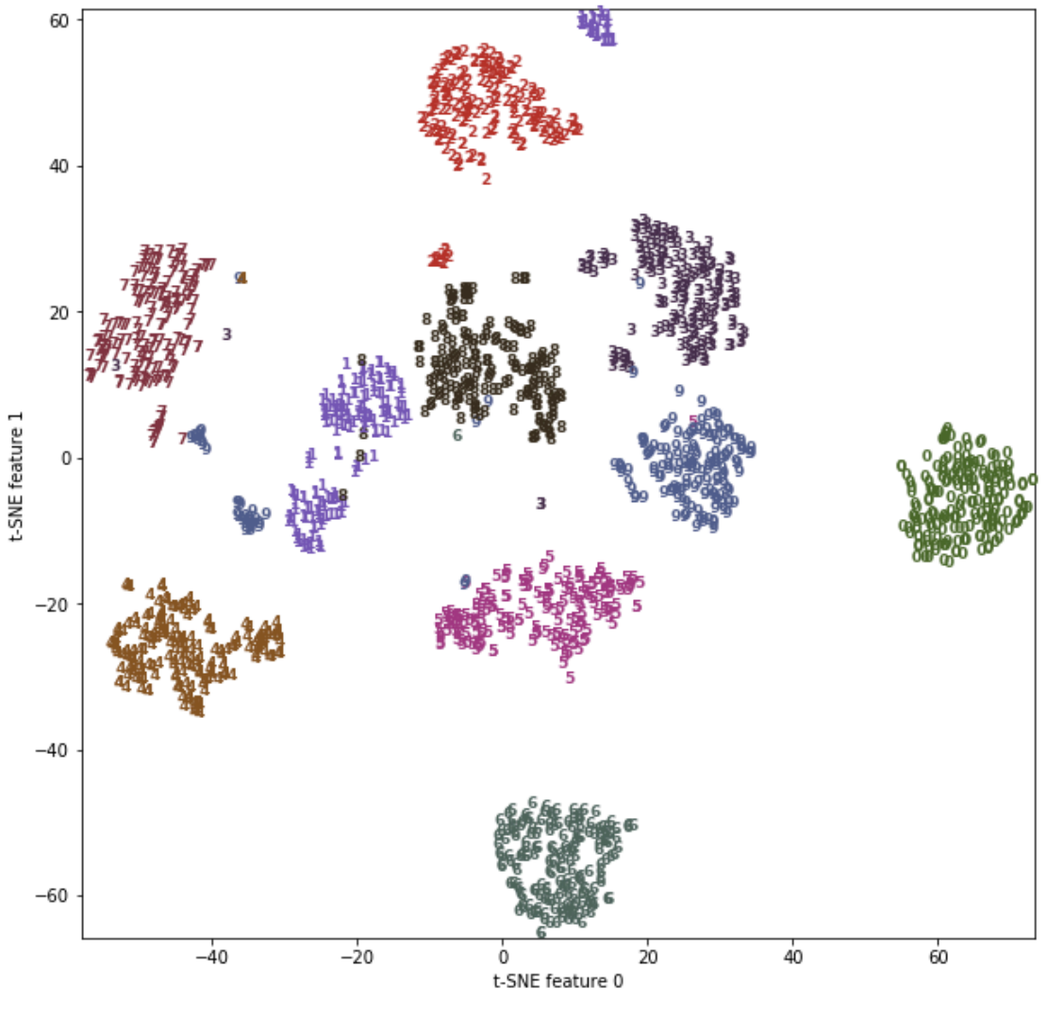

Figure 3-22. Scatter plot of the digits dataset using two components found by t-SNE

Muller, Andreas C. and Sarah Guido. (2016). Introduction to Machine Learning with Python, O'Reilly Media, Inc.

Manifold Learning

- **Best Practices**
  - If the number of dimensions is > 50, reduce to 50 dimensions by first running PCA
- **Parameters**
  - Perplexity - number of nearest neighbors considered, generally larger data sets require larger perplexity
- **Strengths**
  - Retains local distance relationships in the data
- **Weaknesses**
  - Can’t be applied new data - won’t work for transforming a training set and then applying the same transformation to the testing set
  - Bases values can vary based on starting point and number of dimensions
  - No closed form solution