Again mean normalization ensures that every attribute or feature of the dataset has a zero mean, while feature scaling ensures all the features are within the same numeric range.

Finally, PCA is susceptible to vary wildly due to slight perturbations or changes in the dataset.

PCA with Scikit-learn

In this section, PCA is implemented using Scikit-learn.

```
# import packages
from sklearn.decomposition import PCA
from sklearn import datasets
```

from sklearn.preprocessing import Normalizer

```
# load dataset
data = datasets.load_iris()

# separate features and target
X = data.data

# normalize the dataset
scaler = Normalizer().fit(X)
normalize_X = scaler.transform(X)

# create the model.
pca = PCA(n_components=3)

# fit the model on the training set
pca.fit(normalize_X)

# examine the principal components percentage of variance explained
pca.explained_variance_ratio_

# print the principal components
pca_dataset = pca.components_
pca_dataset
'Output':
```

```
array([[ 0.18359702,  0.49546167, -0.76887947, -0.36004754],
       [ 0.60210709, -0.64966313, -0.05931229, -0.46031175],
       [-0.2436305 ,  0.28528504,  0.49319469, -0.78486663]])
```

In this chapter, we explained PCA giving a high-level overview of how it works to find a low-dimensional sub-space of a dataset. More so, we showed how PCA is implemented with Scikit-learn. This chapter concludes Part 4. In the next part, we introduce another scheme of learning methods called deep learning that builds on the machine learning neural network algorithm for learning complex representations.

PART V

Introducing Deep Learning