Introduction to Machine learning

Stijn Kammer (S4986296) Ramon Kits (S5440769)

September 21, 2022

1 Introduction

1.1 The assignment

In this report was written for the the first assignment of Introduction to Machine Learning. This report will look into the methods used for the assignment this week, after which it will go over the experimental results and finishing off with a discussion chapter.

2 Methods

This section will cover the methods used in this report

2.1 The PCA Algorithm

The PCA algorithm is a algorithm to help reduce the dimensionality of large data sets. It is defined as an othogonal linear transformation. Using eigenvectors it will select the n dimensions that will give the most information. These n dimensions we can discard the others to reduce the dimensionality.

2.2 Implementation

Python Implementation PCA

```
import matplotlib.pyplot as plt
import scipy.io as sio
import numpy as np

# load COIL20 dataset from matlab file
coil20 = sio.loadmat('COIL20.mat')
# extract data and labels
data = coil20['X']
labels = coil20['Y']

# perform PCA
# calculate the mean of each column
mean = np.mean(data, axis=0)
# subtract the mean from each column
data = data - mean
# calculate the covariance matrix
cov = np.cov(data, rowvar=False)
# calculate the eigenvalues and eigenvectors of the covariance matrix
eigval, eigvec = np.linalg.eig(cov)
# sort the eigenvalues and eigenvectors in descending order
idx = eigval.argsort()[::-1]
eigval = eigval[idx]
eigvec = eigvec[:,idx]
# select the first two eigenvectors
eigvec = eigvec[:,0:2]
# project the data onto the eigenvectors
data_pca = np.dot(data, eigvec)
```

2.3 PCA output

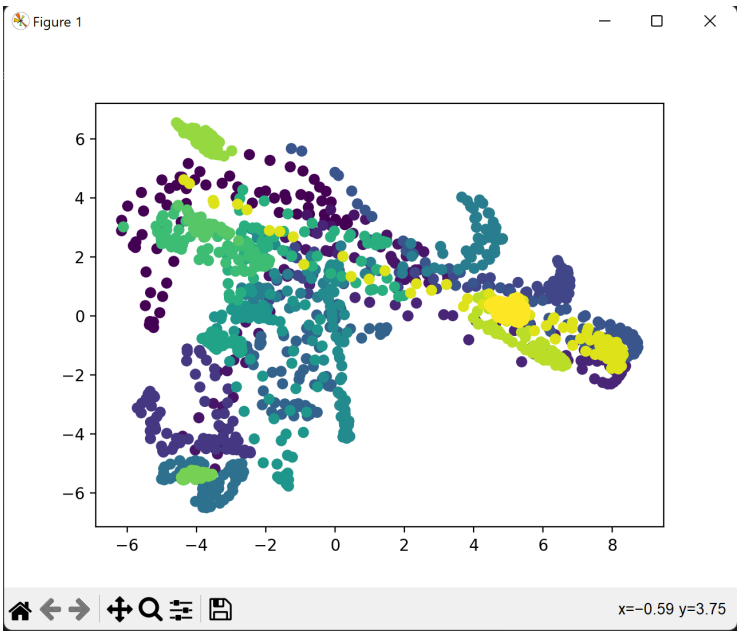


Figure 1: Output of the PCA on dataset X.

3 Experimental result

3.1 Eigen-value profile

Using the PCA algorithm a visual view of the data was created. For this view two eigen values have been used. In figure 2 the distribution of importance of the different eigenvalues has been visualized.

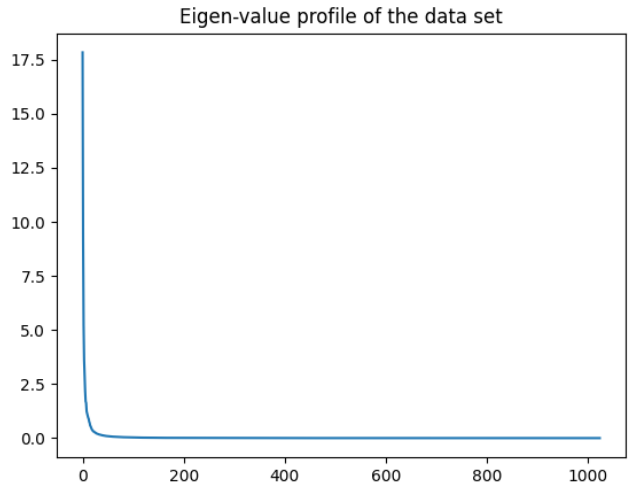


Figure 2: Eigen value profile of X

3.2 Dimensionality needs for fractions of the variance

This section will go over the dimensionality needed for 0.9, 0.95 and 0.98 fractions of the total variance.

In figure 3 is visible that the dimensionality needed for 0.9 fractions of the total variance is 39, 0.95 needs a dimensionality of 83 and 0.98 a dimensionality of 174.

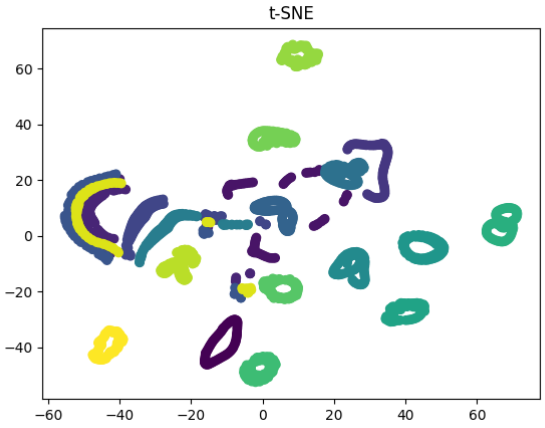
d	0.9	0.95	0.98
1024	39	83	174

Figure 3: Dimensionality needed for 0.9, 0.95, 0.98 fractions of the total variance.

3.3 t-SNE data reduction

t-SNE is a nonlinear dimensionality reduction technique that can be used to visualize high-dimensional data in a way that is easy to understand by a data analyst or viewer. This is very usefull because there is not a straight forward way of visualizing four or more dimensions in one figure. t-SNE analyzes the data and keeps the highest variance possible so clusters stay apparent.

In figure 4 the reduction to a two-dimensional representation of the data is shown.



(a) t-SNE datapoints

(b) original images

Figure 4: Principal components of X.

4 Discussion

4.1 t-SNE

The t-SNE algorithm shows an interesting pattern on the left side of of figure 4(a) where a it shows three values which show a large similarity, these values are the three pictures of the cars.

4.2 Eigen-value profile

For the Eigen-value profile of the data set a few values have a significantly bigger impact over the majority.

5 Individual workload

For this first assignment we worked with duo coding, in this fashion we worked equally and efficiently on the code. As for the report, some time was spend individually learning how LaTeX worked after which we worked on the different subsections of the report individually by equally sorting and splitting them by the amount of work required. After which we reviewed the sections of the other.