MATH38161 Multivariate Statistics and Machine Learning Coursework

Task A: Dimension reduction for MNIST data using principal component analysis (PCA)

Jack Hodgkinson
Student ID: 10727774

2023-12-01

Dataset

The Modified National Institute and Standards and Technology (MNIST) database, is a collection of handwritten digit images used extensively in character recognition, image classification and machine learning research [Deng, 2012]. Lecun et al. [1998] created the database from the NIST Special Database 3 (SD-3), a collection of handwritten digits from Census Bureau employees, and the Special Database 1 (SD-1), a collection of handwritten digits from high school students. The MNIST dataset is a mixture of the SD-3 and SD-1 databases, hence the name modified-NIST or MNIST. Furthermore, the MNIST digits were size-normalised to a 28x28 pixel image by computing the centre of mass of the pixels and translating the images to position the point at the center of the 28x28 pixel field.

Figure 1: The first 5 images of the MNIST database

The dataset that will be analysed in this coursework is a subset of the full MNIST data, which consists of 10,000 images. The subset that this analysis will be run on is the test data of the full MNIST database. Through exploratory analysis, I found that each image is a handwritten digit representing an integer in the range of 0 to 9, and that each pixel in the respective image contains a value between 0 and 255. The collection of pixels is what creates the image we can see. Figure 1 displays the first 5 images of the MNIST test sample that will be used in this coursework.

Methods

Principal Component Analysis (PCA) is a dimension reduction technique aimed at identifying a reduced set of features that represent the original data in a lower-dimensional subspace whilst retaining variation through minimising information loss [Kherif and Latypova, 2020]. PCA can deliver an overview about the most important variables that contribute the most to the difference and similarities between variables [Groth et al., 2013].

The PCA transformation involves applying orthogonal decomposition to data in order to ensure that all the resulting components are orthogonal [Laing et al., 2002]. PCA was first discovered in statistics by Pearson in 1908, when he formulated the analysis as part of research into orthogonal regression. Then in the 1930s, Hotelling further developed the work of Pearson to formulate PCA as we know it today [Wold et al., 1987].

From the lecture notes, it can be assumed that we start with a random vector $\boldsymbol{x}$ with $\text{Var}(\boldsymbol{x}) = \boldsymbol{\Sigma} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathbf{T}}$. Then, we apply an orthogonal transformation of the original components of $\boldsymbol{x}$ to produce:

$$\underbrace{\boldsymbol{t}^{\text{PCA}}}_{\text{Principal components}} = \underbrace{\boldsymbol{U}^T}_{\text{Orthogonal matrix}} \boldsymbol{x}$$

$$\text{where } \text{Var}(\boldsymbol{t}^{\text{PCA}}) = \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_d \end{pmatrix}$$

As seen above, the principal components do not have unit variance hence PCA cannot be considered as a whitening procedure.

Applying PCA to a data matrix $\boldsymbol{X}$ differs slightly, as now after the transformation we get $\boldsymbol{T}$, which is the sample version of principle components. This can be represented by the formula below:

$$\boldsymbol{T} = \boldsymbol{X}\boldsymbol{U}$$

Although we have discussed that we need the orthogonal matrix $\boldsymbol{U}$ to calculate the principal components, how do we actually compute $\boldsymbol{U}$? We can find $\boldsymbol{U}$ through one of the methods described below. Note that $\boldsymbol{X}_c$ denotes the column centered data matrix, meaning that all the components in $\boldsymbol{X}$ have been centered through multiplying $\boldsymbol{X}$ by the centering matrix $\boldsymbol{C} = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}_{n\times n}$

1. **Eigenvalue Decomposition using the empirical estimate for covariance matrix:** Estimate the covariance matrix by $\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\boldsymbol{X}_c^T\boldsymbol{X}_c$ and then apply eigenvalue decomposition on $\hat{\boldsymbol{\Sigma}} = \hat{\boldsymbol{U}}\Lambda\hat{\boldsymbol{U}}$ to find $\boldsymbol{U}$.
2. **Singular Value Decomposition on the centered data matrix:** Use the singular value decomposition of $\boldsymbol{X}_c = \boldsymbol{V}\boldsymbol{D}\boldsymbol{U}^T$ to find $\boldsymbol{U}$.

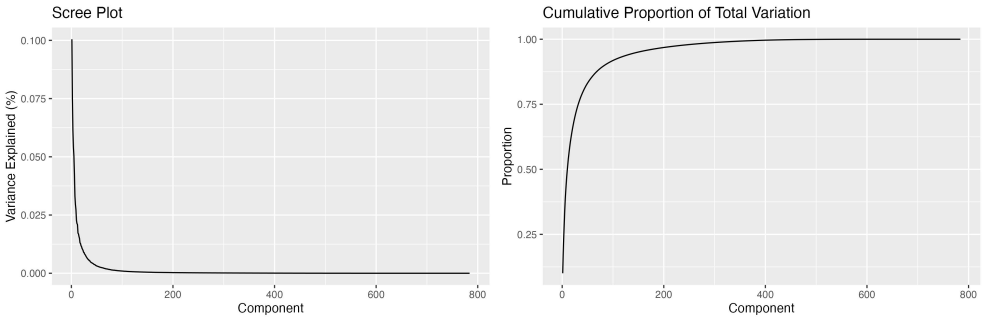
For the purpose of this coursework, I will be completing the PCA transformation on the MNIST data in R using the **prcomp()** function. This function centers the input data matrix and then uses the singular value decomposition to find $\boldsymbol{U}$ by default. For all R code, please refer to the **Appendix**.

Results and Discussion

Before we discuss the results of running PCA on the MNIST dataset, it is important to note that there has been no scaling, outlier removal or normalisation of the dataset before PCA has been performed.

From running PCA on the MNIST dataset, we now have 784 principal components which have been computed from the original 784 pixel variables. These components have been computed so that the first principal component captures the most variability in the data as possible, then the second and so on. This can be seen on the scree plot in the figure below:

Figure 2: Scree and Cumulative Proportion of Total Variation Plots

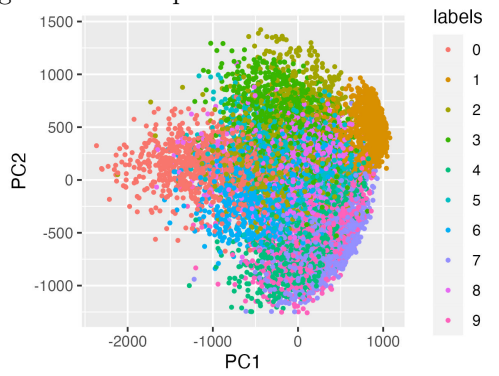


The scree plot on the left shows that the first principal component explains around 10% of the variance. Then, the line shows that the variance explained decreases for each subsequent principal component, with the *elbow* around 50 principal components. The elbow can be defined as the point after which the variance explained decreases in a linear fashion, and we sometimes use the elbow rule as a method of identifying which components to retain [Shilaskar and Ghatol, 2013]. This pattern can also be seen on the Cumulative Proportion of Variation plot to the right. Using the **find_curve_elbow** function from the **pathviewr** package in R, it can be calculated that the elbow of the explained variance is at the $48th$ principal component.

An alternative way of deciding the number of components to use is by setting a threshold of cumulative explained variance, and then selecting the number of components that generate that cumulative sum of

explained variance. Often, this threshold is 90% or 95% of the cumulative proportion of total variation. Through exploring the PCA-transformed MNIST data in R, it can be said that 148 principal components capture 95% of the variation. From this, one could concluded that a suitable dimension reduction could be from 784 to 148, as we would still be capturing 95% of the variation whilst having a lower dimension dataset, meaning the dataset would be easier to analyse and reduce processing time.

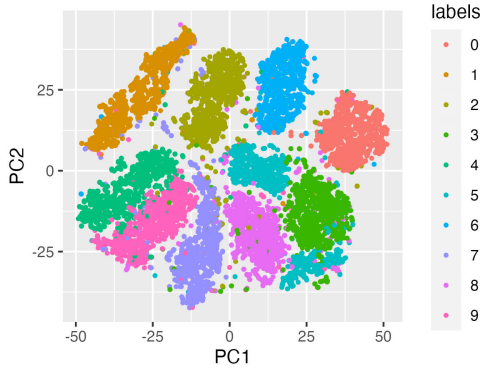
Figure 3: Scatterplot of PC1 vs PC2 after PCA



As can be seen from the scatterplot of PC1 and PC2 to the left, there are no obvious clusters for each label. Although it appears that the components labelled 1 are grouped together, the rest of the plot features much overlap. For example, looking at the bottom right of the plot, there is a large group of components with the labels 4, 7 and 9 all overlapping. To explain the majority of the cumulative proportion of total variance, we require much more than 3 principal components, making visualising the clusters in this case very difficult. This is why the scatterplot does not provide a lot of information regarding the clustering of the data. Whilst Ding and He [2004] proved that PCA dimension reduction automatically performs data clustering according to the K-means objective function, the scatterplot does not visualise this due to the large amount of dimensions.

Another approach to dimension reduction resulting in clearer clustering could be to use the **t-SNE** algorithm. **T-Distributed Stochastic Neighbour Embedding** is a non-linear probabilistic technique which finds an optimal way to project data points into a lower dimensional space such that clustering of these data points is similar as were in high-dimensional space. T-SNE maintains the internal structure of the data whilst projecting it to a lower dimensional space [Pareek and Jacob, 2021]. This means that clusters are much more visible after running t-SNE on the data rather than PCA. This is evident from the scatterplot to the right, which was produced from results created by running the **Rtsne()** function in the **Rtsne** package on the MNIST data in R. As can be seen, the majority of the data is clustered into groups, with minimal data points lying outside the clusters.

Figure 4: Scatterplot of PC1 vs PC2 after t-SNE



In conclusion, whilst PCA is successful for dimension reduction for the MNIST data, it is unsuccessful for visualising clusters as the amount of principal components needed to explain the majority of the cumulative proportion of total variation exceeds visible dimensions. However, using a non-linear algorithm such as t-SNE is results in more successful cluster visualisation for high dimensional data. t-SNE is the algorithm that I personally would use in the future for dimensionality reduction.

Appendix

```
# Load the packages required.
library(broom)
library(dplyr)
library(ggplot2)
library(magrittr)
```

```r
library(patchwork)
library(pathviewr)
library(purrr)
library(Rtsne)
library(tidyr)

# Exploratory Data Analysis.
load("mnistTest.rda")
dim(mnistTest$x)
range(mnistTest$x)
mnistTest$y[1:5]

# Generate the first 5 images of MNIST dataset and save for figure.
jpeg("images.jpg", width=1000, height=200)
par(mfrow = c(1,5), pty="s")
generate_digit <- function(i){
  m =  matrix(mnistTest$x[i,] , nrow=28, byrow=TRUE)
  image(t(apply(m, 2, rev)), col=grey(seq(1,0,length=256)), axes = FALSE)}
digits <- purrr::map(1:5, generate_digit)
dev.off()

# Computing 784 principal components from the 784 original pixel variables.
labels <- mnistTest$y
pca_output = prcomp(mnistTest$x)

# Producing Scree Plot and Cumulative Variation Plot.
princ_comps_var <- broom::tidy(pca_output, matrix="pcs")

plot1 <- ggplot(princ_comps_var, aes(x=PC, y=percent)) +
  geom_line() +
  labs(x = "Component",
       y = "Variance Explained (%)",
       title = "Scree Plot")

plot2 <- ggplot(princ_comps_var, aes(x=PC, y=cumulative)) +
  geom_line() +
  labs(x = "Component",
       y = "Proportion",
       title = "Cumulative Proportion of Total Variation")

# Find Elbow of Explained Variance
princ_comps_var %>%
  dplyr::select(PC, percent) %>%
  pathviewr::find_curve_elbow()

# Calculating how many components explain 95% of the variation.
princ_comps_var %>%
  filter(cumulative <= 0.95) %>%
  count()

# Produce scatterplot of PC1 and PC2 after PCA.
princ_comps <- broom::tidy(pca_output, matrix ="u") %>%
  pivot_wider(names_from=PC, values_from=value, names_prefix="PC") %>%
```

```
  dplyr::select(-1)

plot3 <- ggplot(princ_comps, aes(x=PC1, y=PC2)) +
  geom_point(aes(colour=labels), size = 0.7)

# Produce scatterplot of PC1 vs PC2 after t-SNE
rtsne_out <- Rtsne(mnistTest$x)
pc_rtsne <- as.data.frame(rtsne_out$Y)
plot4 <- ggplot(pc_rtsne, aes(x = V1, y = V2)) +
  geom_point(aes(colour=labels), size = 0.7) +
  labs(x = "PC1", y = "PC2")

# Save each of the plots to use as figures.
ggsave("pcaplots.jpeg", plot1 + plot2, width = 12, height = 4, dpi = 300, units = "in")
ggsave("scatterplot.jpeg", plot3, width = 4, height = 3, dpi = 300, units = "in")
ggsave("tsneplot.jpeg", plot4, width = 4, height = 3, dpi = 300, units = "in")
```

Bibliography

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi: 10.1109/MSP.2012.2211477.

Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 29, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015408. URL https://doi.org/10.1145/1015330.1015408.

Detlef Groth, Stefanie Hartmann, Sebastian Klie, and Joachim Selbig. *Principal Components Analysis*, pages 527–547. Humana Press, Totowa, NJ, 2013. ISBN 978-1-62703-059-5. doi: 10.1007/978-1-62703-059-5_22. URL https://doi.org/10.1007/978-1-62703-059-5_22.

Ferath Kherif and Adeliya Latypova. Chapter 12 - principal component analysis. In Andrea Mechelli and Sandra Vieira, editors, *Machine Learning*, pages 209–225. Academic Press, 2020. ISBN 978-0-12-815739-8. doi: https://doi.org/10.1016/B978-0-12-815739-8.00012-2. URL https://www.sciencedirect.com/science/article/pii/B9780128157398000122.

Y. C. Laing, H.P. Lee, S.P. Lim, W.Z. Lin, K.H. Lee, and C.G. Wu. Proper orthogonal decomposition and its applications—part i: Theory. *Journal of Sound and Vibration*, 252(3):527–544, 2002. ISSN 0022-460X. doi: https://doi.org/10.1006/jsvi.2001.4041. URL https://www.sciencedirect.com/science/article/pii/S0022460X01940416.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Jyoti Pareek and Joel Jacob. Data compression and visualization using pca and t-sne. In *Advances in Information Communication Technology and Computing: Proceedings of AICTC 2019*, pages 327–337. Springer, 2021.

Swati Shilaskar and Ashok Ghatol. Dimensionality reduction techniques for improved diagnosis of heart disease. *International Journal of Computer Applications*, 61(5), 2013.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37–52, 1987. ISSN 0169-7439. doi: https://doi.org/10.1016/0169-7439(87)80084-9. URL https://www.sciencedirect.com/science/article/pii/0169743987800849. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.