MT5758

Multivariate Analysis

University of St Andrews

SCHOOL OF MATHEMATICS AND STATISTICS

Assignment

Typographical Analysis Using MNIST

Produced by:

Ifeanyi Anthony Omeifejideofor (220024321)

Group 4.

1. INTRODUCTION

The introduction of language has played a role for mankind to communicate in a variety of ways, early writing systems such as hieroglyphics enabled the first great civilization to maintain records, govern their people and pass on their cultural achievements to future generations but as the decades eroded, language has evolved drastically, thus connecting people, societies, and countries. In the evolution of modern language, humans became more idea-driven and technologically advanced owing to the standardized nature of modern languages, which convey a shared understanding of written symbols, enabling humanity to express, communicate, transmit, and preserve knowledge, creativity, and culture.

Typography is the art and technique of arranging, enhancing words and symbols to make written language more legible and visually appealing, it has shaped the modern semantics of language, especially written words and grammar, it's development has been aided by technology hence affecting and changing how the marks we perceive as characters are made and presented (ambrose et al,.2006), furthermore, serving as a pivotal force and determinant in the progression and transformation of linguistic expression and communication. Many typographers understand the reading process and are similarly concerned with single letter recognition, The renowned typographer Walter Tracy defined legibility as being the "clarity of single characters", following this designation, issues such as character definition, contrast, stroke angle, weight, width, resolution, and hinting can all be influencing legibility (Beier et al.,2010). The importance of legibility cannot be overemphasized as this has been a point of interest for many researchers especially for dyslexic patients who may have difficulties recognising and reading specific words, O'Brien (2005) demonstrated that by enhancing typographical aspects like the size and contrast of print, individuals with dyslexia could exhibit reading patterns similar to skilled readers. Specifically, larger print sizes enabled dyslexic patients to achieve their maximum reading speed, compared to non-dyslexic readers.

In recent years as society continues to head toward digitalization, legibility has become increasingly vital for the development of intelligent character recognition (ICR) systems. Intelligent character recognition is the task of deciphering digitized handwritten text (Putcha et al,.2018). There are lots of instances where handwritten documents are still very much prevalent such as court proceedings or documents, invoices, taxes, and receipts however this information may need to be digitalized. As vast amounts of information are generated and consumed daily, it is essential for ICR systems to effectively process diverse typographic styles and ensure the seamless digitalization of written content. In order to develop more models capable of recognizing and separating a wide range of textual inputs, researchers must gain a deeper understanding of the factors that influence legibility and typographic enhancements. As a result, robust and adaptive systems will be able to handle the ever growing demands of information processing.

This paper seeks to uncover details regarding typography and legibility to answer the following

- Can typographic enhancement improve legibility?
- Can machine learning models be utilized to decipher and separate digitalized text?
- Can we understand effectiveness of our typographic enhancements. And rank them accordingly?

2. METHODS

The data utilized in this analysis is the MNIST dataset, The MNIST dataset was created by LeCun and others at AT&T Laboratories in 1998. It was made by combining two NIST (National Institute of Standards and Technology) datasets - Special Databases 1 and 3 - which are made up of images of digits handwritten by high school students and US Census Bureau employees respectively. These images were converted to a 28x28 pixel image that matches MNIST. It contains 70,000 centred images of fixed size representing digits from 0 to 9. The images are all in grayscale, hence 0 indicates pure black pixels and 255 pure white; values in between represent a shade of grey. Each image's pixels may be flattened into a vector of 784 dimensions, with each component taking an integer value in this range. It is worth noting that the data was already pre-processed hence we did not perform any processing steps to the images.

In order to address the research questions in this study the following steps were undertaken for this analysis.

- Dimensionality Reduction.
- Employing a scree plot for the selection of eigenvectors based on explained variance
- Using Unsupervised models (Clustering) to identify misclassified Values.
- Typographic enhancement of the dataset.
- Ranking the effectiveness of the typographic enhancement

Dimensionality Reduction

The MNIST is a high dimensional dataset that consist of over 70,000 images with 28 x 28-pixel images with a flattened vector of 784 dimensions hence a dimensional reduction technique was required to reduce features in the dataset while retaining the most important feature. The technique used for this analysis was the PCA (Principal Component Analysis). The principal component analysis (PCA) is a mathematical algorithm that reduces the dimensionality of the data while retaining most of the variation in the data set. It accomplishes this reduction by identifying directions, called principal components, along which the variation in the data is maximal. By using a few components, each sample can be represented by relatively few numbers instead of by values for thousands of variables (Ringnér, 2008).

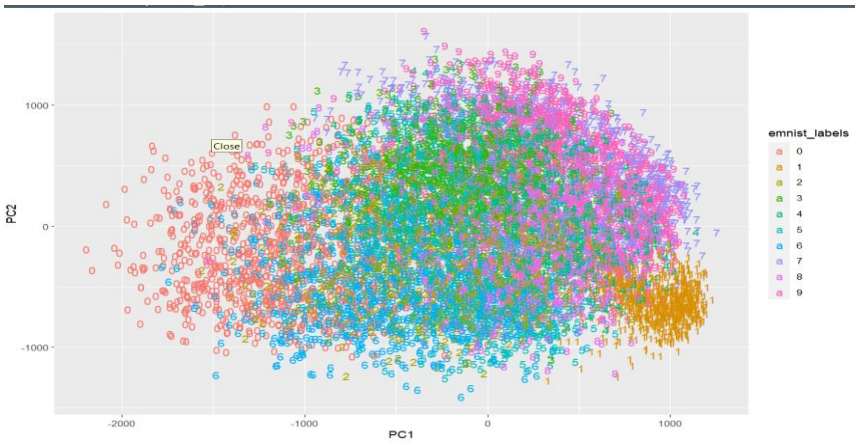

Figure 1: Plot showing dimensionality reduction technique using the PCA.

Scree plot for the selection of eigenvectors based on explained variance.

The scree is a line plot showing principal components of our analysis, it was used to determine the number of principal components to retain that explains optimal variance which captures a significant amount of information in the MNIST dataset.

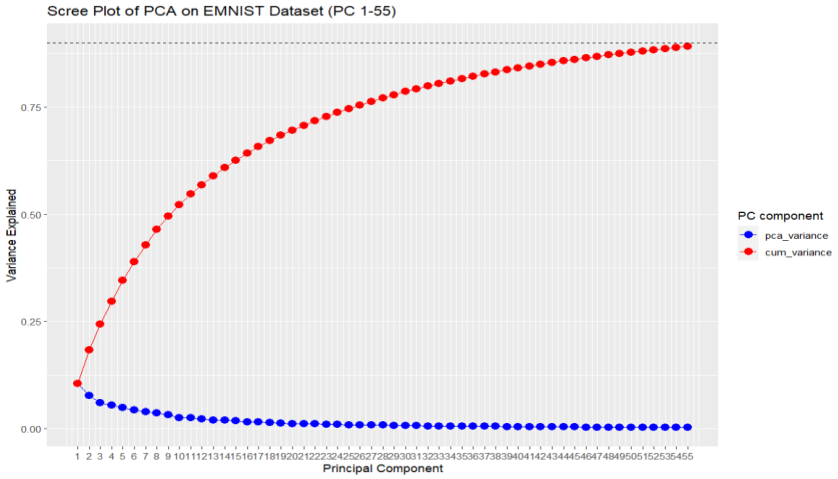

Figure 2: *Plot showing screen plot of principal components. (Colour Red is the cumulative explained variance and colour Blue is the PCA components)*

Using Unsupervised models (Clustering) to identify misclassified Values.

Clustering is a useful tool, It is a method for finding cluster structure in a data set that is characterized by the greatest similarity within the same cluster and the greatest dissimilarity between different clusters (Sinaga et al., 2020). The K-means is an unsupervised machine algorithm used for clustering, which is the process of partitioning the dataset into distinct groups based on patterns or similarities. The k-means algorithm is iterated through necessary conditions for minimizing the k-means objective function with updating equations for cluster centres and memberships, respectively, as

$$a_k = \frac{\sum_{i=1}^{n} z_{ik} x_{ij}}{\sum_{i=1}^{n} z_{ik}} \text{ and}$$

$$z_{ik} = \begin{cases} 1 & \text{if } \|x_i - a_k\|^2 = \min\limits_{1 \le k \le c} \|x_i - a_k\|^2 \\ 0, & \text{otherwise.} \end{cases}$$

where ‖xi−ak‖ is the Euclidean distance between the data point xi and the cluster centre ak (Sinaga et al, 2020). After applying PCA to the dataset, the dimensionality-reduced data was fed into the K-means algorithm, using 10 clusters. Following the partitioning process, an analysis was conducted to identify the digits with the highest misclassification rates. This information was then used to determine the appropriate digits and the specific typographic enhancements to be applied to them.

Typographic Enhancement of the Dataset

Several typographic enhancements were made to some of the images in the MNIST dataset which are I) adding a horizontal slash across 7 ii) adding a dot inside a zero iii) adding a horizontal slash to the number 4 to form a foot.

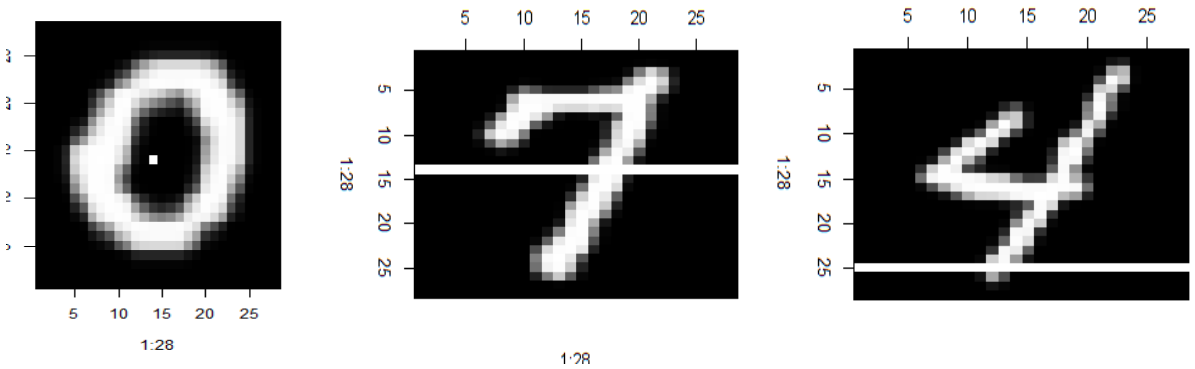

Figure 3: *Images showing typographic enhancements to Numbers zero, seven and four.*

The enhancement of Numbers seven and four were done by first extracting the image matrix from the dataset, then calculating the row for the horizontal stroke and then updating the image matrix with the white stroke while for digit zero, the centre of the image was calculated and the corresponding pixel value was set to white. The updated image matrix was saved back to the dataset.

Ranking effectiveness of the typographic enhancement

To assess the impact of typographic enhancements on the legibility of digit clustering, we performed a comparative analysis of the K-means model accuracy using the original and enhanced datasets. The evaluation was carried out by computing the mean value of each cluster and dividing it by the total number of clusters. If the mean score increased after the enhancement, it would suggest an improvement in clustering and better legibility, while a decreased score would indicate that the enhancements had no effect on legibility. We also used BCSS (Between-Cluster Sum of Squares) as a performance metric to measure the difference between clusters and the amount of variation between them. BCSS helped to quantify the effect of typographic enhancements on clustering performance. Finally, a bar plot was used to visualize the digit clustering before and after the enhancements, which provided useful insights for both visualization and performance assessment purposes.

3. RESULTS

In the beginning, the model faced challenges in categorizing digits appropriately within the initial cluster using the original dataset. The score was computed by identifying the mode of each cluster and dividing it by the total count within that cluster. For instance, in the first cluster, digit 5 emerged as the mode, appearing 383 times. This value was then divided by the combined sum of misclassified instances in other clusters, with the overall average obtained

by dividing the result by 10. The model's performance on the original dataset exhibited a 57% accuracy rate, which is marginally better than random guessing. Following typographic improvements, the model's performance rose to 67%, reflecting a 10% increase. This indicates that the typographical analysis contributed to better digit readability when utilizing ICR systems like K-means.

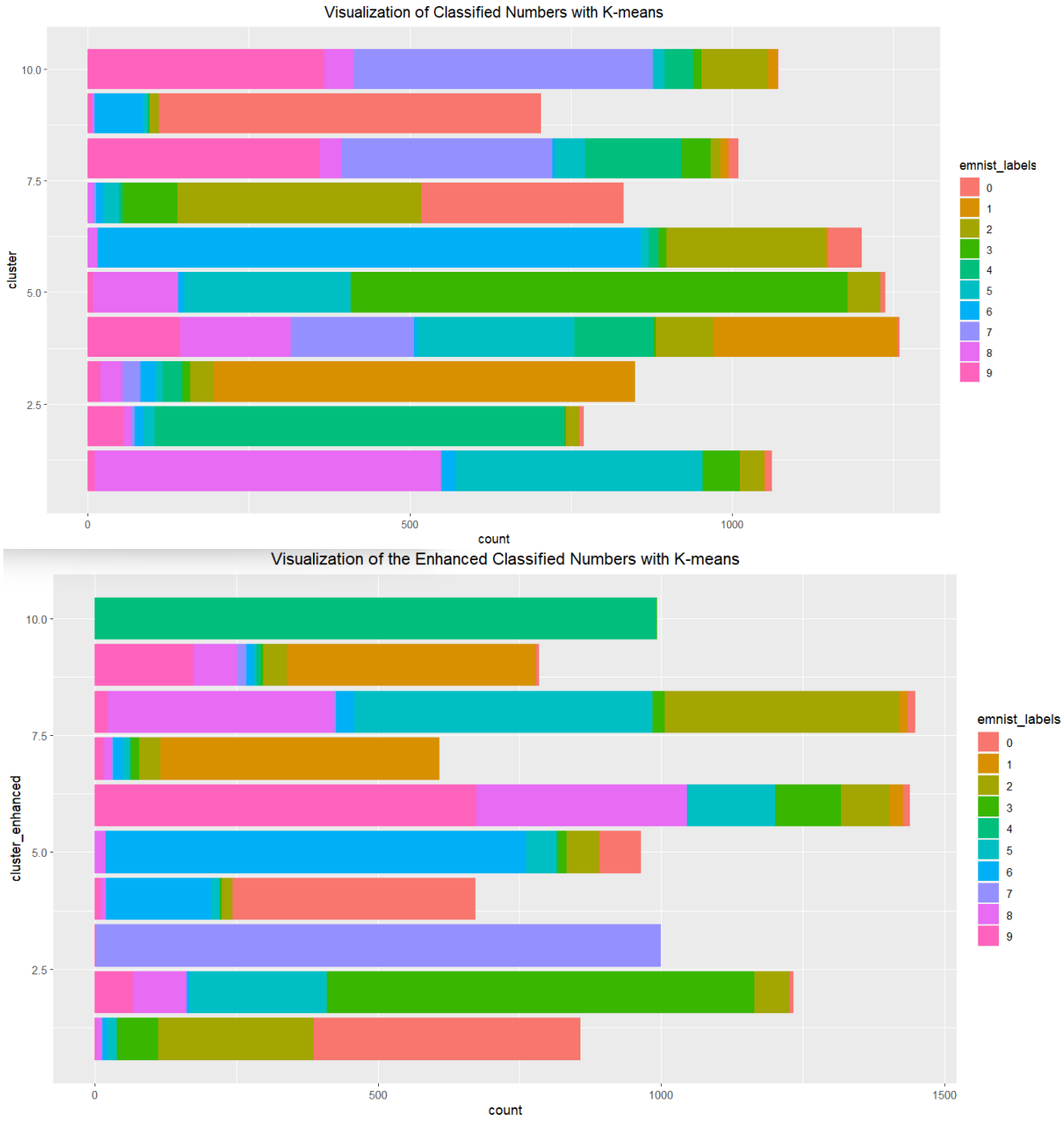

Figure 4: *The figure displays the upper image representing the K-means classification of the initial dataset, while the lower image showcases the classification of the improved dataset.*

The illustration reveals that following dataset enhancement, the classification accuracy for the digits four and seven exceeded 90%, In espouse The model successfully identified digit 4 with 99.8% accuracy and digit 7 with 99.6% accuracy. However, the least effective

typographic enhancement was the addition of a dot to the digit 0, with the model struggling and achieving only 56.1% accuracy.

Typographic Enhancement	Effectiveness Rank	Justification	Accuracy
Adding a foot to 4	Most Effective.	Correctly classified	99.8%
Adding a slash across 7	Effective.	Correctly classified with some errors.	99.6%
Adding a dot in 0	Least Effective.	Model struggled with classification	56.1%

Table 1: *Illustrating the effectiveness of the typographic enhancement.*

Both the BCSS and WCSS were analysed to evaluate data variability in K-means clustering pre- and post-enhancement. The BCSS showed a substantial 14.28% improvement, indicating better-separated values in the enhanced dataset. Meanwhile, the WCSS experienced a more modest change, with a 1.08% increase in the enhanced dataset compared to the original. This highlights that while the enhanced dataset resulted in a marked improvement in separating values, the overall clustering quality exhibited a less pronounced increase.

	BCSS Enhanced Dataset	BCSS Original Dataset	BCSS % Change	Accuracy Original Dataset	Accuracy enhanced dataset	% change in accuracy
K-means clustering	11,846,815,083.	10,365,361,093.	14.28%	57%	67%	10%

	WCSS Enhanced dataset	WCSS Original dataset	% change in WCSS
K-means clustering	25,928,494,515	25,650,270,186	1.08%

Table 2: *Table illustrating the impact of typographic enhancements.*

4. CONCLUSION

The findings demonstrate that typographic enhancement plays a crucial role in improving legibility, which is particularly important given the worldwide need for developing intelligent character recognition systems to identify and digitize written text, as well as in healthcare fields such as understanding dyslexia and enhancing readability. Future research should concentrate on further typographic enhancements and exploring their impact on legibility. This would contribute to the development of more precise models for processing textual content and advancing digitization efforts.

REFERENCES

Beier, S. and Larson, K. (2010) *Design improvements for frequently misrecognized letters* , *Research Gate*. Available at: https://www.researchgate.net/publication/233608849_Design_Improvements_for_Frequently_Misrecognized_Letters (Accessed: April 1, 2023).

O'Brien, B.A., Mansfield, J.S. and Legge, G.E. (2005) *The effect of print size on reading speed in dyslexia*, *Journal of research in reading*. U.S. National Library of Medicine. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1427019/#R41 (Accessed: April 1, 2023).

Ptucha, R. *et al.* (2018) *Intelligent character recognition using fully convolutional neural networks*, *Science Direct*. Pergamon. Available at: https://www.sciencedirect.com/science/article/pii/S0031320318304370 (Accessed: April 1, 2023).

Ringnér, M. (2008) *What is principal component analysis?*, *Nature News*. Nature Publishing Group. Available at: https://www.nature.com/articles/nbt0308-303 (Accessed: April 1, 2023).

Sinaga, K. and Yang, M.-S. (2020) *Unsupervised K-means clustering algorithm - IEEE XPLORE*, *IEEE Explore*. Available at: https://ieeexplore.ieee.org/abstract/document/9072123 (Accessed: April 1, 2023).

Appendix

2023-04-03

```
#Reading the csv file with our model
# Load necessary libraries
library(dplyr)
library(ggplot2)
library(pracma)
library(cluster)
library(factoextra)


#Read the data set and convert it to a data frame
emnist_balanced <- read.csv('emnist-balanced-train.csv')

#Renaming the columns in our the label column in the data frame
emnist_balanced <- emnist_balanced%>%rename('label' = X45)

#Checking for the unique labels in the emnist data frame
unique_values <- unique(emnist_balanced['label'])


#remove the small letter alphabets since there are not complete on the
balanced data set
emnist_balanced_new <- select(emnist_balanced, everything()) %>% filter(label
<= 9)

set.seed(123)
#Randomly selecting 10,000 of the original image
emnist_shuffled <- sample(nrow(emnist_balanced_new), 10000)
emnist_shuffled <- emnist_balanced_new[emnist_shuffled, ]
emnist_labels <- as.factor(emnist_shuffled$label)


# Making Enhancement to the Images 7
add_white_slash <- function(images) {
  labels <- images$label

  for (i in 1:nrow(images)) {
    if (labels[i] == 7) {
      # Add a white horizontal stroke across the image
      image_matrix <- images[i, -1] # removing the first column (label) from
the row
      image_dim <- sqrt(length(image_matrix))
      image_matrix <- matrix(image_matrix, nrow = image_dim, ncol =
image_dim) # Reshape to original dimensions
```

```
      # Determine the row for the horizontal stroke
      stroke_row <- floor(image_dim / 2)

      # Make the row white
      image_matrix[stroke_row, ] <- 255

      # Update the images data.frame
      images[i, -1] <- as.vector(image_matrix)
    }
  }

  return(images)
}


# Making Enhancement to the Images 4
feet_added_number_four <- function(images) {
  labels <- images$label

  for (i in 1:nrow(images)) {
    if (labels[i] == 4) {
      # Add a white horizontal stroke across the image
      image_matrix <- images[i, -1] # removing the first column (label) from
the row
      image_dim <- sqrt(length(image_matrix))
      image_matrix <- matrix(image_matrix, nrow = image_dim, ncol =
image_dim) # Reshape to original dimensions


      # Make the row white
      image_matrix[25:25.5, ] <- 255

      # Update the images data.frame
      images[i, -1] <- as.vector(image_matrix)
    }
  }

  return(images)
}


# Making Enhancement to the Images 0
zero_added_dot <- function(images) {
  labels <- images$label

  for (i in 1:nrow(images)) {
    if (labels[i] == 0) {
      # Add a white horizontal stroke across the image
```

```
      image_matrix <- images[i, -1] # removing the first column (label) from
the row
      image_dim <- sqrt(length(image_matrix))
      image_matrix <- matrix(image_matrix, nrow = image_dim, ncol =
image_dim) # Reshape to original dimensions

      #Calculate the centre of mass for our images
      center_x = (ncol(image_matrix) + 1)/2
      center_y = (nrow(image_matrix) + 1)/2

      #Drawing a dot on the image
      image_matrix[center_x, center_y] <- 255

      # Update the images data.frame
      images[i, -1] <- as.vector(image_matrix)
    }
  }

  return(images)
}



#Adding a horizontal slash across the Number 7
enhanced_emnist_balanced <- add_white_slash(emnist_shuffled)

#Adding a feet across the Number 4
enhanced_emnist_balanced<- feet_added_number_four(enhanced_emnist_balanced)

# Adding a dot across the Number Zero
enhanced_emnist_balanced<- zero_added_dot(enhanced_emnist_balanced)


#Visualize random random samples of our emnist data set
#Plot an EMNIST image
plot_emnist <- function(data){

  #Create a sample of images to be plotted
  sample_img <- sample(nrow(data), 8)

  #create a 2x2 plot
  par(mfrow =c(2, 4))

  #create a for loop to loop through our sample of images
  for(i in 1:length(sample_img)){
    #choosing a random sample of an image
    img <- data[sample_img[i], 2:785]
    label <- data[sample_img[i], 1]
```

```
    #Reshape the image into a 28 by 28 matrix
    image_matrix <- matrix(img, nrow =28, ncol = 28)

    #convert the images into matrix of numbers since it is in character
    image_matrix_numbers <- apply(image_matrix, 2, as.numeric)

    #plot the image
    image(1:28, 1:28, image_matrix_numbers,  col = grey((0:255)/255))
    title(paste('label:', label))}
}

#Viewing the original Image
plot_emnist(emnist_balanced_new)

#Viewing the enhanced Images for Number 7
plot_emnist(enhanced_emnist_balanced)



#Dropping the label column for the original data set
emnist_columns <- select(emnist_shuffled, 2:785)

#Dropping the label column for the original data set
enhanced_columns <- select(enhanced_emnist_balanced, 2:785)


#performing PCA on the original image to reduce dimensions of the data set.
emnist_pca <- prcomp(emnist_columns)

enhanced_pca <- prcomp(enhanced_columns)

#creating a plot to visualize PCA
OG_data_pca_plot <- ggplot(as.data.frame(emnist_pca$x), aes(x=PC1, y=PC2,
colour = emnist_labels, label = emnist_label))+
                    geom_text(aes(label=emnist_labels))

OG_data_pca_plot


# Compute variance explained by each principal component
pca_variance_explained <- emnist_pca$sdev^2 / sum(emnist_pca$sdev^2)

# Compute cumulative variance explained
cumm_variance <- cumsum (pca_variance_explained)

# Create a scree plot for the first 10 principal components
scree_plot <- ggplot(data = data.frame(PC = 1:55,
```

```
                                        Variance_Explained =
pca_variance_explained[1:55],
                                        Cumulative_Variance_Explained =
cumm_variance[1:55]),
                     aes(x = PC, y = Variance_Explained)) +
  geom_point(size = 3, aes(color = "pca_variance")) +
  geom_line(aes(color = "pca_variance")) +
  geom_point(aes(y = Cumulative_Variance_Explained, color = "cum_variance"),
size = 3) +
  geom_line(aes(y = Cumulative_Variance_Explained, color = "cum_variance")) +
  xlab("Principal Component") +
  ylab("Variance Explained") +
  ggtitle("Scree Plot of PCA on EMNIST Dataset (PC 1-55)") +
  scale_x_continuous(breaks = seq(1, 55, by = 1)) +  # Set x-axis range to 1-
30
  geom_hline(yintercept = 0.9, linetype = "dashed", color = "black") + # Add
abline at 90%
  scale_color_manual(name='PC component',
                     breaks=c('pca_variance', 'cum_variance'),
                     values=c('pca_variance'='blue', 'cum_variance'='red'))

scree_plot

#Creating a data frame to store the score (Eigen vector) of the reduced data
till the 55th PC component
reduced_data <- data.frame(emnist_pca$x[, 1:55])
reduced_data_with_labels <- cbind(emnist_labels, reduced_data)

#Fitting a k-means to identify wrongly classified values
#The K-means did not converge in 10 iterations so iterations is increased to
15 maximum
k_means <- kmeans(reduced_data, centers = 10, iter.max = 15)

#Plotting a Scatter plot for the K-means clustering
# Criteria to select number of clusters using within cluster sum of squares
within_cluster_plot <- fviz_nbclust(x = reduced_data, FUNcluster = kmeans,
iter.max = 40,
                                    method = "wss", k.max = 20)
within_cluster_plot

#Plot to visualize K-means Clustering
cluster <- k_means$cluster
reduced_data_with_labels <- cbind(cluster, reduced_data_with_labels)

# Plot to visualize K-means Clustering
cluster_plot <- ggplot(data = reduced_data_with_labels, aes(y = cluster)) +
  geom_bar(aes(fill = emnist_labels)) +
  ggtitle('Visualization of Classified Numbers with K-means') +
  theme(plot.title = element_text(hjust = 0.5))
```

```
cluster_plot

#Between Cluster sum of Square for the original data set
wcss_original <- sum(k_means$withinss)
tss_original <- sum(var(reduced_data) * (nrow(reduced_data) - 1))

bcss_original <- tss_original - wcss_original
bcss_original


# Calculating the Proportion of Misclassified text.
cluster_one <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 1)
table(cluster_one$emnist_labels)
prop_one <-
max(table(cluster_one$emnist_labels))/sum(table(cluster_one$emnist_labels))

cluster_two <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 2)
table(cluster_two$emnist_labels)
prop_two <-
max(table(cluster_two$emnist_labels))/sum(table(cluster_two$emnist_labels))


cluster_three <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 3)
table(cluster_three$emnist_labels)
prop_three <-
max(table(cluster_three$emnist_labels))/sum(table(cluster_three$emnist_labels
))


cluster_four <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 4)
table(cluster_four$emnist_labels)
prop_four <-
max(table(cluster_four$emnist_labels))/sum(table(cluster_four$emnist_labels))

cluster_five <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 5)
table(cluster_five$emnist_labels)
prop_five <-
max(table(cluster_five$emnist_labels))/sum(table(cluster_five$emnist_labels))

cluster_six <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 6)
table(cluster_six$emnist_labels)
prop_six <-
```

```
max(table(cluster_six$emnist_labels))/sum(table(cluster_six$emnist_labels))


cluster_seven <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 7)
table(cluster_seven$emnist_labels)
prop_seven <-
max(table(cluster_seven$emnist_labels))/sum(table(cluster_seven$emnist_labels
))


cluster_eight <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 8)
table(cluster_eight$emnist_labels)
prop_eight <-
max(table(cluster_eight$emnist_labels))/sum(table(cluster_eight$emnist_labels
))

cluster_nine <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 9)
table(cluster_nine$emnist_labels)
prop_nine <-
max(table(cluster_nine$emnist_labels))/sum(table(cluster_nine$emnist_labels))

cluster_ten <- select(reduced_data_with_labels, everything()) %>%
filter(cluster == 10)
table(cluster_ten$emnist_labels)
prop_ten <-
max(table(cluster_ten$emnist_labels))/sum(table(cluster_ten$emnist_labels))


#calculating the weighted average of the KNN Score.
KNN_score <- sum(prop_one + prop_two + prop_three + prop_four + prop_five +
                   prop_six + prop_seven + prop_eight + prop_nine +
prop_ten)/10

KNN_score


#Creating a data frame to store the score (eigen vector) of the reduced data
till the 55th PC component
reduced_data_enhanced <- data.frame(enhanced_pca$x[, 1:55])
k_means_enhanced <- kmeans(reduced_data_enhanced, centers = 10, iter.max =
15)

cluster_enhanced <- k_means_enhanced$cluster

reduced_data_enhanced_label <- cbind(emnist_labels, reduced_data_enhanced)
reduced_data_enhanced_label <- cbind(cluster_enhanced,
```

```
reduced_data_enhanced_label)

# Plot to visualize K-means Clustering on the enhanced data set
cluster_plot_enhanced <- ggplot(data = reduced_data_enhanced_label, aes(y =
cluster_enhanced)) +
  geom_bar(aes(fill = emnist_labels)) +
  ggtitle('Visualization of the Enhanced Classified Numbers with K-means') +
  theme(plot.title = element_text(hjust = 0.5))

cluster_plot_enhanced

# Calculating the Proportion of Misclassified text.
cluster_one_enhanced <- select(reduced_data_enhanced_label, everything()) %>%
filter(cluster_enhanced == 1)
table(cluster_one_enhanced$emnist_labels)
prop_one_enhanced <-
max(table(cluster_one_enhanced$emnist_labels))/sum(table(cluster_one_enhanced
$emnist_labels))


cluster_two_enhanced <- select(reduced_data_enhanced_label, everything()) %>%
filter(cluster_enhanced == 2)
table(cluster_two_enhanced$emnist_labels)
prop_two_enhanced <-
max(table(cluster_two_enhanced$emnist_labels))/sum(table(cluster_two_enhanced
$emnist_labels))


cluster_three_enhanced <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced  == 3)
table(cluster_three_enhanced$emnist_labels)
prop_three_enhanced <-
max(table(cluster_three_enhanced$emnist_labels))/sum(table(cluster_three_enha
nced$emnist_labels))


cluster_four_enhanced <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced == 4)
table(cluster_four_enhanced$emnist_labels)
prop_four_enhanced <-
max(table(cluster_four_enhanced$emnist_labels))/sum(table(cluster_four_enhanc
ed$emnist_labels))

cluster_five_enhanced <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced == 5)
table(cluster_five_enhanced$emnist_labels)
prop_five_enhanced <-
max(table(cluster_five_enhanced$emnist_labels))/sum(table(cluster_five_enhanc
ed$emnist_labels))
```

```
cluster_six_enhanced <- select(reduced_data_enhanced_label, everything()) %>%
filter(cluster_enhanced == 6)
table(cluster_six_enhanced$emnist_labels)
prop_six_enhanced <-
max(table(cluster_six_enhanced$emnist_labels))/sum(table(cluster_six_enhanced
$emnist_labels))


cluster_seven_enhanced  <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced == 7)
table(cluster_seven_enhanced$emnist_labels)
prop_seven_enhanced <-
max(table(cluster_seven_enhanced$emnist_labels))/sum(table(cluster_seven_enha
nced$emnist_labels))


cluster_eight_enhanced <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced == 8)
table(cluster_eight_enhanced$emnist_labels)
prop_eight_enhanced <-
max(table(cluster_eight_enhanced$emnist_labels))/sum(table(cluster_eight_enha
nced$emnist_labels))

cluster_nine_enhanced <- select(reduced_data_enhanced_label, everything())
%>% filter(cluster_enhanced == 9)
table(cluster_nine_enhanced$emnist_labels)
prop_nine_enhanced <-
max(table(cluster_nine_enhanced$emnist_labels))/sum(table(cluster_nine_enhanc
ed$emnist_labels))

cluster_ten_enhanced <- select(reduced_data_enhanced_label, everything()) %>%
filter(cluster_enhanced == 10)
table(cluster_ten_enhanced$emnist_labels)
prop_ten_enhanced <-
max(table(cluster_ten_enhanced$emnist_labels))/sum(table(cluster_ten_enhanced
$emnist_labels))



#calculating the weighted average of the KNN Score.
KNN_score_enhanced <- sum(prop_one_enhanced + prop_two_enhanced +
prop_three_enhanced + prop_four_enhanced +
                            prop_five_enhanced + prop_six_enhanced +
prop_seven_enhanced +
                            prop_eight_enhanced + prop_nine_enhanced +
prop_ten_enhanced)/10

KNN_score_enhanced
```

```
#Between Cluster sum of Square for the enhanced data set
wcss_enhanced <- sum(k_means_enhanced$withinss)
tss_enhanced <- sum(var(reduced_data_enhanced) * (nrow(reduced_data_enhanced)
- 1))

bcss_enhanced <- tss_enhanced - wcss_enhanced
bcss_enhanced
```