# HANDWRITTEN DIGIT RECOGNITION

## PROJECT 4 – DATA MINING – Clustering

David Dessommes
SMU Lyle School of Engineering
EMIS 7332 – Data Mining for Analytics
May 2, 2016

## Abstract

This fourth project continues to apply the concepts of Data Mining to a practical application and a real data set. In this case, a sample image data set of handwritten digits is analyzed via Clustering, in order to understand concepts of feature extraction and how they are instrumental in the analysis of Data Mining problems. Clustering via the k-Means, Hierarchical and DBSCAN methods were attempted, with k-Means resulting in more understandable results. Within Sum of Squares (WSS) and Average Silhouette Width (ASW) measures were used to compare different cluster selections. Based on this project's results, additional research in feature extraction mechanisms, algorithms and Cluster Analysis may be needed to increase the validity of Clustering on a problem data set such as this one.

Correspondence concerning this paper can be sent to David Dessommes via email to daviddessommes@yahoo.com

TABLE OF CONTENTS

# INTRODUCTION

This project focuses on the application of Cluster Analysis techniques on an image-based numbers data set. The numbers data set contains the numerical representation of handwritten digits on a 28x28 pixel grid, with the value of each pixel in the grayscale from 0-256. The underlying approach or methodology to be used is the CRISP-DM process for Data Mining efforts.

The goal at this juncture is to identify features and clustering methods that may lend themselves better to resolving this problem. This project leveraged the code and references provided during class lecture and did web research to find feature extraction methods on similar problems. The Cluster Analysis seen in class were used and discussed in this report. Conclusions about the robustness and applicability of the feature extraction and cluster analysis method are made.

# DATA PREPARATION

## Preprocessing and Feature Selection

Trier, Jain and Taxt (1996) did a survey on relevant feature extraction methods. They argue "that there is only a limited number of independent features that can be extracted from a character image". They also highlight the importance and dependency with the preprocessing stages of the analysis. Preprocessing will typically consist of: "(a) Binarization (…) using a global or a locally adaptive method; (b) Segmentation (…) (c) An (optional) conversion to another character representation" (Trier, Jain and Taxt, 1996). Note that "The task of Binarization [sic] itself is necessary since most commercial recognition algorithms work only on binary images since it proves to be simpler to do so." (Wikipedia - OCR, 2016).

The feature extraction methods surveyed, which are applicable to Gray scale sub-image representations are: ***template matching, deformable templates, unitary transforms, zoning, geometric moments, Zernike moments***. Other methods applicable solely to either binary or vector representations are: *projection histograms, contour profiles, discrete features, spline curve and Fourier descriptors* (Trier, Jain and Taxt, 1996).

- **Template Matching** "is a technique (…) for finding small parts of an image which match a template image" (Template Matching, 2016). A cursory search in R did not reveal any implementation for this technique. The Wikipedia reference does outline an implementation algorithm.
- **Unitary (Image) Transforms** are methods, which for the purposes of feature extraction, "obtain a reduction of features while preserving most of the information about the character shape" (Trier, Jain and Taxt, 1996). The survey investigated the Karhunen-Loeve (KL), Fourier, Hadamard and Haar transforms, with KL being the most effective but at a high computational cost. Cosine, sine and slant transforms were also analyzed where "the Cosine transform has been coined the method of choice for image data compression" (Trier, Jain and Taxt, 1996). In R, the packages that appear to have some limited application of Fourier transforms related to the problem at hand are s*eewave* and

*waveslim*. However because of uncertainty of application and time constraints, these R packages were not explored further.

- **Zoning** refers to the computation of average gray levels for zones within the image (Trier, Jain and Taxt, 1995). Another definition of zoning compares it with layout analysis, which is more along the lines of document character recognition (OCR, 2016).
- **Image Moments** "The use of moment invariants as features for pattern recognition have been extensively used." (…) "Zernike moments have been used (…) for character recognition of binary solid symbols (…) experiments suggest that they are well suited for gray scale character sub-images as well" (Trier, Jain and Taxt, 1996).

  > An image moment is a certain particular weighted average (moment) of the image pixels' intensities, or a function of such moments, usually chosen to have some attractive property or interpretation. Image moments are useful to describe objects after segmentation. Simple properties of an image which are found via image moments include area, (…) centroid, and information about its orientation. (Image Moment, 2016)

  R packages that, upon cursory search, appear to support relevant calculations upon image vectors are *imager*, *IM*, and *ripa*. Interesting introductory sites for describing Image moments, with less intimidating math than Wikipedia page, are found in the References under Analysis of Binary Images (1997) and Statistical Moments (2002).

- **Projection Histograms** Are discussed in Trier, Jain and Taxt, (1996) for binary images but more recent source fail to mention them. No R packages appear to mention, upon cursory search, Projection Histograms for purposes of Image Analysis.

Other methods described in the references noted so far were not explored further. Giuliodori, Lillo and Peña (2011) discuss the Hough transform, but a search of R packages only found one library *PET* for a specific tomography image application. Wikipedia (Feature detection, 2016) discusses a wide range of methods for feature extractions. However less than a handful of R packages appear to support any of these functions. The library *adimpro* supports edge detection using Laplacian, Sober or Robert Cross filter. The libraries *spatialfil*, *smoothie* appear to have direct applicability to image processing and 2D convolution in particular Laplacian of Gaussian (LoG). The library *kernlab* appears to have the basic operations that could be further developed but would require additional development. On a final note regarding R

libraries for feature selection. The R library biOps appeared to readily support a large part of the operations seen in class, but has been discontinued in CRAN for some time now. It appears the functionality has been ported over to the *imager* library.

For the problem at hand, it is useful to find different features as the pixel columns themselves may not lend themselves to appropriate clustering analysis. "Choosing the right features is still very important. Better features can produce simple and more flexible models, and they often yield better results" (Wikipedia – Feature Engineering, 2016).

For this project, Table 1 shows the features that were extracted from the numbers data set comprised of pixel intensity counts of the scanned grayscale digit images.

| COLUMN NAME | DEFINITION | COMMENTS |
|---|---|---|
| pixelSUM_ALL | Sum All Pixel Values | |
| pixelSUM_H1 | Sum Pixel Values (0-391) | |
| pixelSUM_H2 | Sum Pixel Values (392-783) | |
| pixelSUM_Q1 | Sum Pixel Values (0-195) | |
| pixelSUM_Q2 | Sum Pixel Values (196-391) | |
| pixelSUM_Q3 | Sum Pixel Values (392-587) | |
| pixelSUM_Q4 | Sum Pixel Values (588-783) | |
| pixelPCT_H1 | pixelSUM_H1 / pixelSUM_ALL | RETAINED |
| pixelPCT_H2 | pixelSUM_H2 / pixelSUM_ALL | RETAINED |
| pixelPCT_Q1 | pixelSUM_Q1 / pixelSUM_ALL | RETAINED |
| pixelPCT_Q2 | pixelSUM_Q2 / pixelSUM_ALL | RETAINED |
| pixelPCT_Q3 | pixelSUM_Q3 / pixelSUM_ALL | RETAINED |
| pixelPCT_Q4 | pixelSUM_Q4 / pixelSUM_ALL | RETAINED |
| pixelAVG_ALL | Avg All Pixel Values | RETAINED |
| pixelAVG_H1 | Avg Pixel Values (0-391) | |
| pixelAVG_H2 | Avg Pixel Values (392-783) | |
| pixelAVG_Q1 | Avg Pixel Values (0-195) | |
| pixelAVG_Q2 | Avg Pixel Values (196-391) | |
| pixelAVG_Q3 | Avg Pixel Values (392-587) | |
| pixelAVG_Q4 | Avg Pixel Values (588-783) | |
| pixelSUM_Horiz | Sum of Pixels from Horizontal Line Convolution | RETAINED |
| pixelSUM_Vert | Sum of Pixels from Vertical Line Convolution | RETAINED |
| pixelSUM_Sobel | Sum of Pixels from Sobel Filter Convolution | RETAINED |
| pixelAVG_CentX | Avg pixel value along X axis (Centroid) | RETAINED |
| pixelAVG_CentY | Avg pixel value along Y axis (Centroid) | RETAINED |

**Table 1.** Data Description Summary for Project 4 selected features

The author realized during iterative clustering runs that including all the features above was reducing the modeling effectiveness. In the final rounds only a subset of the features were retained as noted in the Comments column of Table 1. The R libraries that assisted in feature selection were IM (Rajwa, Dundar, Irvine & Dang, 2013) and spatialfil (Dinapoli and Gatta, 2015).

## MODELING

The **stats** library from R statistical package was used to analyze the data by means of k-means and hierarchical clustering methods (R, 2016). R package **seriation** (Hahsler, Buchta & Hornik, 2008, 2016) was used to visually compare the cluster validity of the data and visually inspect the digit image elements.

Libraries Rcpp (Eddelbuettel, 2013) and inline (Sklyar et. al, 2015) were used to execute the 2D convolution function as was provided in class. Library **dplyr** (Wickham & Francois, 2015) was extensively used for data manipulation. Although **dbscan** (Hahsler, 2015) was ran for clustering purposes, it provided more use for this project in outlier detection.

The following high-level steps describe the execution of cluster analysis this project.

- **Sampling was extensively used to run the models**
    - With some exceptions, most code was run on samples between 500 and 5000 image vectors. Given the time constraints most code blocks had to run between 15 and 30 minutes most so little time could be afforded to run some models on all 42000 images, although kmeans() did and could run fast enough for this size.
- **Clustering Tendency was analyzed**
    - Using the visual capabilities of *seriation* package, it could be seen that the raw data did not lend itself easily to clustering.
- **Post Feature Extraction, K-Means Clustering was used extensively to determine optimum k for WSS (Within Sum of Squares) and ASW (Average Silhouette Width)**
    - It is worth noting that, easily, over 50% of the effort in this project went into understanding and attempting to extract useful features with which to perform clustering on this data. Notwithstanding after this was done, k-Means was

executed multiple times over samples to determine any noticeable reduction in WSS and well as analyze the ASW.

- **Implement alternative processing to see the effects, if any, in improving the WSS and ASW**
    - Given the fact that Internal Cluster statistics were lower than anticipated in explaining the variance, 3 analysis vectors were explored independently on top of the existing features: a) Would **thinning** all images during pre-processing help? b) Would Principal Components Analysis (**PCA**) help? c) Would "aggressive" **outlier removal** help?
- **Perform tests against the truth table for external validation**
    - Somewhat analogous to Classification Models where sets are split into Training and Testing, the analysis can verify whether the models are grouping data elements according to their true grouping.

## Cluster Determination

As mentioned earlier, tendency in the data to cluster was analyzed. As you can see from Figure 1 below, no noticeable cluster patterns appear in sample (n = 500) taken from the data set.

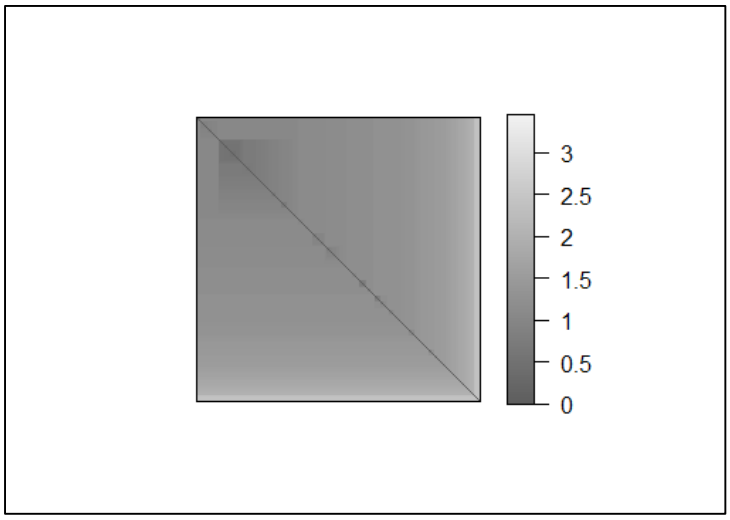


**Figure 1** iVAT from sample (n=500) of digit numbers

This led to the Feature Extraction research and implementation work described earlier. Most features were scaled as required by clustering techniques to within (-3, 3). Remaining features were in the (0, 1) range, which the analyst does not believe to be a concern.

Without going into too much details, iterations of the k-means algorithm fluctuated between 10 and 30 clusters. The WSS plot analyses in some cases (e.g. Figure 2 below) went up to k = 50, mainly due to the fact that the "knee" could not easily be found. In spite of multiple attempts, the Figure below was the closest that any k-means run got to displaying an actual knee. Expected clusters oscillated between 10 and 30, with ASW optimal points shifting based on the code getting different samples each time. Furthermore, there is no steep drop of WSS somewhere in an expected range based on empirical knowledge of the data, which would indicate features selected not being optimal for clustering.

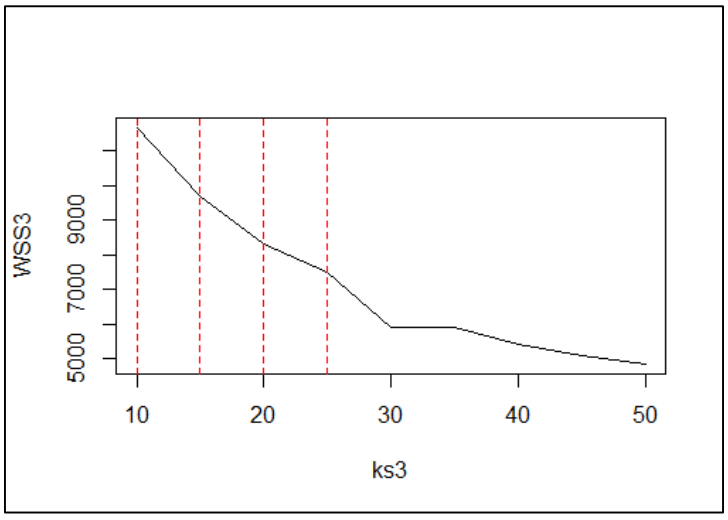


**Figure 2.** Total Within Sum of Squares (WSS) Plot of k-Means scaled (n = 5000)

## Internal Validation Methods

As noted earlier, because WSS plot and initial k-means iterations did not yield expected results, additions were done to the processing to see if results would improve. This included thinning the images, PCA and Outlier Removal (Hahsler, 2016). The ASW plot code seen in class was used for this. The results are in Figures 3-6 on the next page with discussion thereafter.

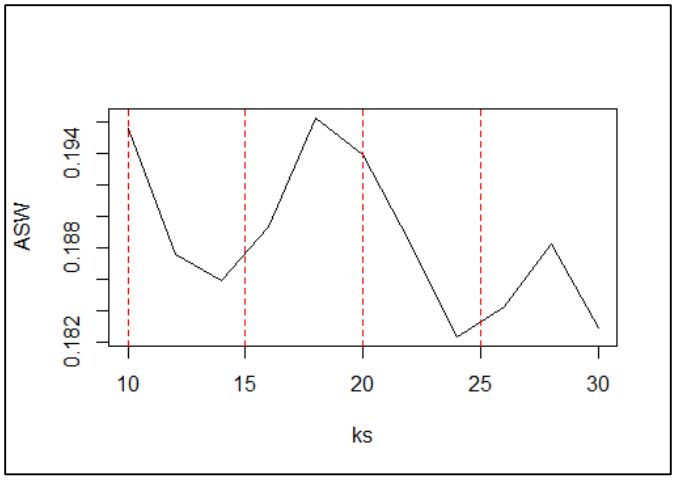


**Figure 3.** Average Silhouette Width (ASW) Plot of k-Means scaled (n = 5000)

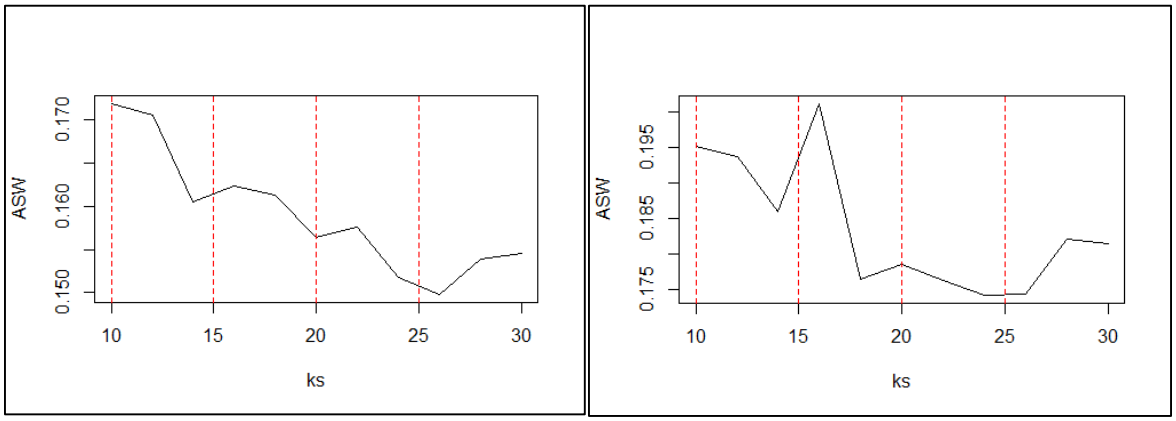


**Figure 4 and 5.** ASW Plot on k-Means scaled (n = 5000) after Thinning (left) and PCA (right)

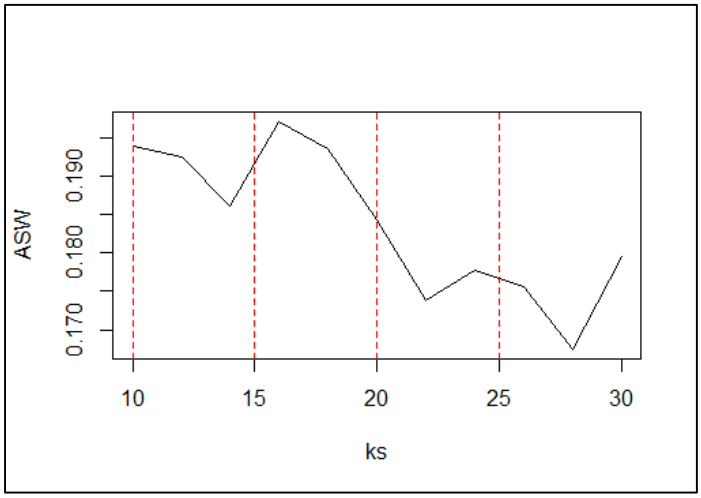


**Figure 6.** ASW Plot on k-Means scaled (n = 5000) after Outlier removal at the sample level

The 2 key features the analyst notices on the ASW plots is a) the range of the y-axis, does it increase significantly? No. b) Are any optimal points/patterns noticeable? Maybe, but the overarching factor is that ASW doesn't significantly increase and that overwrites any patterns in the ASW over k.

## External Validation Methods

R library **fpc** (Hennig, 2015) was used to extract cluster statistics throughout the internal analysis. However we will leverage the library here for external analysis with corrected Rand and VI indicators as seen in class, as well as entropy and purity code (Hahsler, 2016). K-Means Clusters with k = 10, 20 and 30 are compared with randomly generated samples. Unfortunately, but as expected based on the internal validation, the clustering from this project doesn't yield significantly different results from random data.

| | **Corrected Rand** | **VI** | **Entropy** | **Purity** |
|---|---|---|---|---|
| *k-Means 10* | -0.00012 | 4.47 | 2.95 | 0.16 |
| *k-Means 20* | -0.00005 | 5.16 | 1.63 | 0.03 |
| *k-Means 30* | -0.00026 | 5.50 | 1.21 | 0.01 |
| *random 10* | -0.00029 | 4.58 | 2.97 | 0.10 |
| *random 20* | -0.00007 | 5.26 | 1.93 | 0.03 |
| *random 30* | -0.00013 | 5.64 | 1.46 | 0.01 |

**Table 2.** External Validation Summary for Project 4 k-Means vs. Random

## Additional Methods

Hierarchical Clustering and DBSCAN were ran against similar sample sets. DBSCAN was unable to cluster significant groupings above 2 clusters using eps = 0.5 and minPts = 3 based on the kNN distance plot, with n = 1000, 921 points landed in 2 groups only. Regarding HC, the analyst was unable to interpret meaningful patterns from the dendrogram on a comparative sample with k-means (n = 5000).

# EVALUATION AND DEPLOYMENT

The complexity of the problem at the point has not been resolved or even slightly mitigated by the features and analysis done so far. There are some things worth considering as to why?

- Naturally a starting point would be the **feature selection**, an interesting observation is that the code ends up generating a logical matrix (True, False) or (0,1) after using sub-setting code to the image matrix in order to thin the image or extract only a series of dots. This has the perceived advantage of reducing the complexity of the variable. But still a feature must be extracted from the matrix, and this will usually be a summary statistic of the values at each pixel point. In example, what started as a vector whose sum would range from (0, 256x784) upon most feature extraction methods seen in class would result in a vector whose sum would now range from (0, 784). Furthermore, ALL features extracted in this manner or using these operations would yield (0, 784/n), where n is the number of zones into which the image matrix could be conceivable divided into.
- The outlier removal mechanism using LOF from DBSCAN was applied after the sample was taken. Therefore a sample of 5500 elements was taken first, with highest 10% LOF values removed after. Attempts to do first the LOF on the entire data set was resulting time prohibitive based on the time remaining to complete the project. Although in principle, conventional wisdom indicates to the analyst that this should not be of consequence, it is nonetheless documented here for traceability purposes.
- The data was compiled for the specific purposes of classification problems, not clustering. In the end, there has to be some fundamental reason why this problem would be better suited for classification, instead of clustering.

# CONCLUSION

This project has applied Feature Extraction mechanisms and Cluster Analysis methods to a handwritten digits image data set. Class lecture materials and web search results were presented and used as best possible to produce result sets that could be analyzed.

Multiple R libraries were found and used to leverage the Data Mining Project during different stages. The clustering methods seen in class were applied to the features data set. Relevant analysis measures of Within Sum of Squares (WSS) and Average Silhouette Width (ASW) were used to compare the outcomes of the analysis.

The features extracted from this dataset in the manner in which they were extracted did not yield significant results in clustering the data in a favorable fashion when compared to the actual values. This included revised the data set using additional pre-processing mechanism of Thinning as a way of turning the image from grayscale to binary. The use of PCA and Outlier removal did not improve the ASW in any significant manner. External Cluster Validation confirmed that the features extracted and methods use in this particular case did not reveal any significant clustering any closer to the truth cluster when compared to random clusters.

Additional research in Feature Extraction mechanisms, programming algorithms and Cluster Analysis may be needed to increase the validity of these methods on a problem data set as this one.

# REFERENCES


Dinapoli, N. & Gatta, R. (2015). spatialfil: Application of 2D Convolution Kernel Filters to Matrices or 3D Arrays. R package version 0.15. https://CRAN.R-project.org/package=spatialfil

Eddelbuettel, D (2013) Seamless R and C++ Integration with Rcpp. Springer, New York. ISBN 978-1-4614-6867-7.

Giuliodori, A., Lillo, R., & Peña, D. (2011, June). *Handwritten Digit Classification*. Working Paper, Universidad Carlos III de Madrid. Retrieved April 30, 2016 from http://e-archivo.uc3m.es/bitstream/handle/10016/11641/ws111712.pdf;jsessionid=59FB9A728883512A6C328090F5C39E7A?sequence=1

Hahsler, M. (2016). Simple Image Processing. Retrieved May 1, 2016 from http://michael.hahsler.net/SMU/EMIS7332/data/numbers/simple_image_processing.html

Hahsler, M. (2015). dbscan: Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms. R package version 0.9-6. https://CRAN.R-project.org/package=dbscan

Hahsler, M., Buchta C. & Hornik, K. (2016).Infrastructure for seriation. R package version 1.2-0.

Hahsler, M., Hornik, K. & Buchta C. (2008), Getting things in order: An introduction to the R package seriation. Journal of Statistical Software 25/3. URL: http://www.jstatsoft.org/v25/i03/

Hennig, C. (2015). fpc: Flexible Procedures for Clustering. R package version 2.1-10. https://CRAN.R-project.org/package=fpc

R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Rajwa B., Dundar, M., Irvine A. & Tan Dang (2013). IM: Orthogonal Moment Analysis. R package version 1.0. https://CRAN.R-project.org/package=IM

Owens, R. (1997). "Binary Images." Retrieved April 30, 2016, from http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT2/node3.html

Shutler, J. (2002). "Statistical Moments." Retrieved April 30, 2016, from http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/SHUTLER3/CVonline_moments.html

Sklyar, O., Murdoch, D., Smith, M., Eddelbuettel, D., Francois, R., & Soetaert K. (2015). inline: Functions to Inline C, C++, Fortran Function Calls from R. R package version 0.3.14. https://CRAN.R-project.org/package=inline

Trier, O. D., Jain, A. & Taxt, T. (1996) Feature Extraction Methods for Character Recognition. A Survey. *Pattern Recognition*. 29(4): 641–662. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.7439&rep=rep1&type=pdf

Wickham H. & Francois, R. (2015). dplyr: A Grammar of Data Manipulation. R package version 0.4.3. https://CRAN.R-project.org/package=dplyr

Wikipedia: *Feature Detection*. (n.d.). Retrieved April 30, 2016, from https://en.wikipedia.org/wiki/Feature_detection_(computer_vision)

Wikipedia: *Image Moment*. (n.d.). Retrieved April 30, 2016, from https://en.wikipedia.org/wiki/Image_moment

Wikipedia: *Optical Character Recognition*. (n.d.). Retrieved April 30, 2016, from https://en.wikipedia.org/wiki/Optical_character_recognition

Wikipedia: *Template Matching*. (n.d.). Retrieved April 30, 2016, from https://en.wikipedia.org/wiki/Template_matching

## APPENDIX 1 – Code

– The latest version of the code (including version history) of the R script used for this project is publicly available in the location below in case of questions or comments.

https://github.com/ddessommes/digits/blob/master/digits.R

## APPENDIX 2 – HC

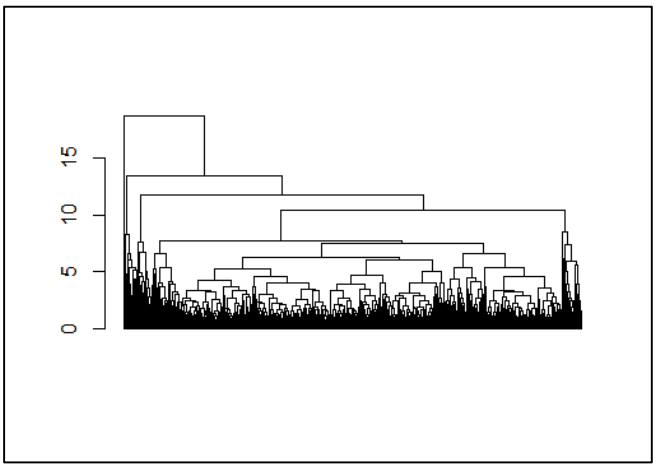


**Figure A2.** HC dendrogram (n = 5000) after Outlier removal, labels removed for readability.