STAT 666 Mini-Project 3

Stephen Merrill and William White

March 30, 2017

Introduction

Every author has a distinct writing style with known characteristics, such as cadence, word choice and perspective, and unknown characteristics such as the number of conjunctions. Because of the recent interest in authorship various statistical methods have been used to analyze text to identify authors based on their writing style. In our analysis we seek to find distinct groups based on writing style through a cluster analysis. We use 18 quantitative measures of writing in order to parse out these writing style groups. We also seek to find ways in which the texts differ from one another with regards to literary style. We then assess whether our writing style groups add an important difference above and beyond the more intuitive super-genre groups.

Methods

In order to appropriately cluster the literary texts, we tested several different methods and made decisions based on a combination of logical reasoning and model performance.

Clustering Methods

As an initial step, we sought to narrow the number of potential clustering algorithms down to those that would provide the best results. We accomplished this through use of the **clValid** package, which compares clustering methods across different amounts of clusters. This comparison is made through the Dunn index and silhouette width, two different measures of compactness within and separation between clusters. We used this **clValid** method to compare results between k-means and agglomerative and divisive hierarchical clustering techniques, with the number of clusters varying from three to seven. Results of this preliminary analysis show that both hierarchical methods outperform k-means but have overlapping results depending on the number of clusters considered. Therefore, we dismissed k-means clustering and proceeded to the next

phase of the model identification process considering only agglomerative and divisive hierarchical clustering methods.

Hierarchical clustering algorithms are sequential processes that, based on a distance between clusters metric, at every step either combine clusters in an agglomerative setting or separate clusters in a divisive setting. As the process continues, groupings of clusters become clear. This can be visualized in tree-like structures called dendrograms.

Linkage Methods

Agglomerative clustering methods vary depending on the definition of distance between clusters that is used. This protocol is known as a linkage method, and is used to determine how items are grouped together. In this study, we initially considered single, complete, average and Ward's method. If there are two clusters, A and B, these linkage methods define distance between clusters as follows.

- Single: Minimum distance between a point in A and B
- Complete: Maximum distance between a point in A and B
- Average: Average distance between all the points in A and B
- Ward's: Distance between clusters is determined by minimization of SSE within the clusters

As a preliminary look at these methods, we built dendrograms using these four agglomerative methods and the divisive method. These visualizations of the clustering algorithms in Figure 1 show that some

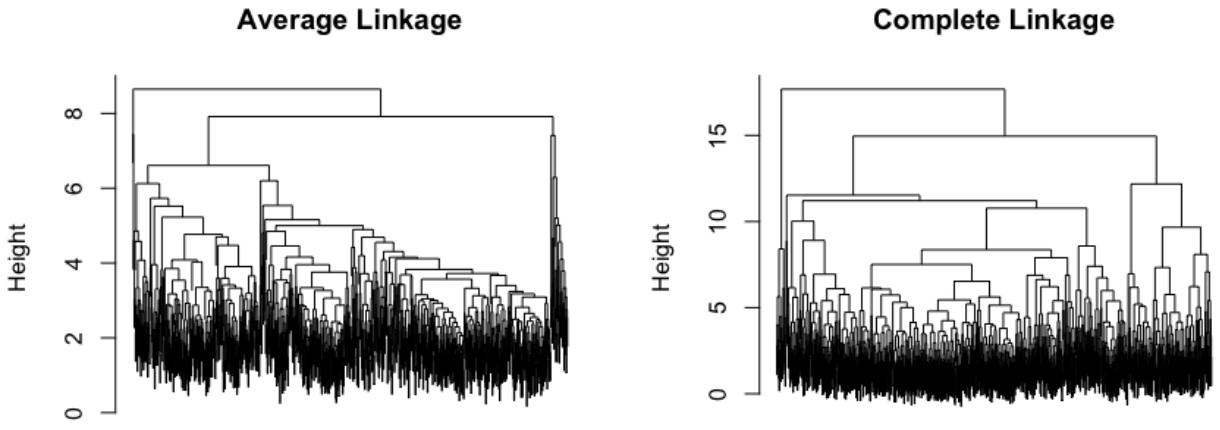

Figure 1: Dendograms from hierarchical clustering, using average and complete linkage

linkage methods do a poor job of creating separate groupings of the texts and should not be considered in our analysis. Specifically, average linkage and single linkage both resulted in dendrograms with long string-like clusters, which create some groups with very few members. Complete linkage, Ward's method, and divisive clustering form dendrograms with a better distribution of members and will better separate the literary texts.

Classification Functions

Having found that divisive, complete linkage and Ward's linkage hierarchical clustering were the best techniques we wanted to assess how well our groups would classify a new text. To accomplish this we treated the cluster labels as true groupings to be used in a classification analysis. We used a holdout method, or cross validation, to estimate the Actual Error Rate (AER) for linear and quadratic classification and K-nearest neighbors (KNN) with K chosen as the square root of the number of observations, about 32. On average, KNN gave us lower AER so for the three clustering techniques we used the KNN AER estimates to choose which cluster method to use. Based on the AER estimates in Table 1 we determined divisive was the best technique with three clusters, Ward's linkage was best for four and five clusters and complete linkage was best for six and seven clusters.

To determine the appropriate number of clusters we initially used the elbow method, which minimizes within cluster SSE and can be seen in in Figure 2. This plot shows that the curve has an "elbow" at three clusters, suggesting that this would be the optimal trade-off point between increasing K and decreasing SSE. To formalize our decision mathematically we used the Gap method[1] to find the number of clusters to include. This method compares the change in within-cluster dispersion with that expected under an appropriate reference null distribution. Figure 2 shows that the Gap statistic has a local maximum at K=3, although it does increase again at higher values. We synthesized these metrics, the AER, and the number of observations put into each cluster and concluded that divisive hierarchical clustering with three clusters is our best approach to cluster the literary texts.

[1] Tibshirani, Walther, Hastie (2000)

# of clusters	3	4	5	6	7
Divisive	**0.047**	0.088	0.104	0.107	0.12
Ward	0.065	0.068	0.094	**0.1**	**0.112**
Complete	0.053	**0.051**	**0.053**	0.113	0.136

Table 1: Estimated AER for divisive hierarchical clustering and agglomerative hierarchical clustering with Ward and complete linkage for different numbers of clusters

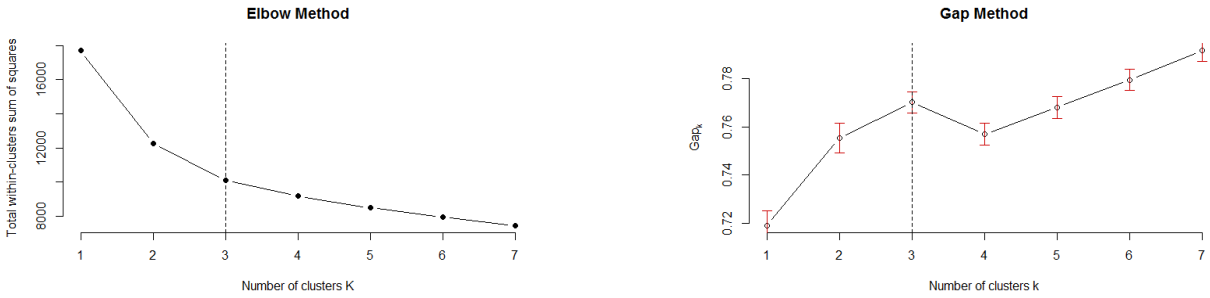

Figure 2: Elbow and Gap analysis indicate that using three clusters is optimal.

Results

Discriminant Function Analysis

Our process of determining the best clustering method for the literary texts yielded divisive hierarchical clustering with three clusters. In order to describe the resulting clusters, we used a discriminant function analysis. The goal of using discriminant functions is to identify the relative contribution of the 18 variables to the separation of the clusters and to find the optimal plane to which the points can be projected to represent the configuration of the clusters. This is done by finding the linear combinations of the 18 variables that maximizes the distance between the three clusters. However, it is not necessary to consider all 18 of the resulting discriminant functions because the separation in the clusters can be explained with far fewer. This concept is called the essential dimension, and is determined as follows. When cluster mean vectors have an essential dimension of only $r < s$ (where s is the dimension of the relevant space), we only need r discriminant functions to describe their separation. r is determined by evaluation a ratio of the sum of the discriminant function eigenvalues.

$$\frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{s} \lambda_i} > c \text{ where } c = .80$$

Our analysis yielded s eigenvalues ($\lambda_1 = 2.644, \lambda_2 = 0.507$), which, using the above ratio, yields an essential dimension of only one as appropriate to describe the separation of cluster means. This implies that there is evidence of collinearity between the cluster means. Discriminant analysis can be extended further by finding standardized coefficients and selecting significantly large values to be interpreted as contributing to group separation. These results can be seen in Table 2. The first essential dimension is therefore interpreted as a relative decrease in the selected covariates will contribute to separation of cluster means.

1st Essential Dimension Variables	Discriminant value
Linear Guidance	-0.604
Word Picture	-0.271
Past Events	-0.265

Table 2: Discriminant functions for the essential dimension

Genre Separation

As a way of evaluating the clustering results, we attempt to characterize each of the three clusters by the genres of each text. These groupings can be seen in Table 3. Our clustering bears out a decent natural separation of genres by cluster. The first and largest cluster contains a high ratio of all types of non-fiction texts, while the second cluster primarily contains fiction texts. The third cluster contains mostly fiction but also bears the presence of "softer" lifestyle non-fiction. We attributed a somewhat ambiguous pleasure-reading name to this cluster. Clearly, the first two clusters are much more interpretable than the last, in part because of the increased number of observations in each cluster.

We also evaluated our cluster results against five pre-defined super-genres. In order to make this comparison, we used our best five-cluster solution, which was hierarchical clustering using complete linkage. The resulting clusters are displayed in Table 4. These results suggest that these super-genres do a poor job of differentiating the literary texts because in some cases they are too specific and in other cases they are too general. For instance, the first cluster produced by our method is best described as a non-fiction, which would be a generalized combination of the first four categories. Meanwhile, the third cluster indicates a specific

	Group 1	Group 2	Group 3
Press: Reporting	79	6	3
Press: Editorial	53	1	0
Press: Reviews	32	0	2
Religion	34	0	0
Skills/Hobbies	61	2	9
Popular Lore	82	7	7
Biography	126	16	8
Official Communication	59	0	1
Learned	159	0	1
Gen Fiction	4	36	18
Mystery	6	36	6
Sci Fi	3	9	0
Adventure/Western	1	43	14
Romance	6	45	7
Humor	8	9	1
Total	713	210	77

Table 3: Three cluster groupings based on genre

	Group 1	Group 2	Group 3	Group 4	Group 5
Press	166	9	0	0	1
Non-Press Non-Fiction	182	6	12	2	0
Biography	141	7	2	0	0
Scholarship/Official Documents	218	1	1	0	0
Fiction	75	112	6	2	57

Table 4: Five cluster groupings compared to super-genres

combination of the super-genres that is difficult to define. Only the fifth cluster seems to fit neatly into a super-genre category. We cannot fully recommend the use of our clustering method over the super-genre distinctions because even though our clustering better separates the groups, it does so with the caveat that our five group solution creates two unidentifiable clusters.

Conclusion

We found the divisive hierarchical clustering with three clusters to be our best approach to group writing styles. We compared different clustering algorithms and linkage methods through metrics, visualization and cross validation to find our best solutions for three to seven clusters. We applied general rules of thumb to metrics used to estimate the optimal number of clusters to include. We then found how our clusters differ from one another using a discriminant analysis and found that a combination of Linear Guidance, Word Picture and Past Events make up the essential dimension that separates the clusters. We believe the improvement in group separation using our clusters is worth the loss of intuitive interpretation which is present with the super-genres. We feel our clusters allow us to see importance differences in writing style that are not apparent when just looking at the super-genre.

R Code

```
library(cluster)
library(clValid)
library(MASS)
cluster<-read.table("~/STAT GRAD/collins.txt",header=TRUE)

####  Initial assessment
test<-clValid(cluster[,2:19],3:7,c("hierarchical","diana","kmeans"),"internal",maxitems = 1200,
method=c("average"))

test2<-clValid(cluster[,2:19],3:7,c("hierarchical"),"internal",maxitems = 1200,method = "single")

test3<-clValid(cluster[,2:19],3:7,c("hierarchical"),"internal",maxitems = 1200,method = "complete")

test4<-clValid(cluster[,2:19],3:7,c("hierarchical"),"internal",maxitems = 1200,method = "ward")

summary(test)
summary(test2)
summary(test3)
summary(test4)

# Hierarchical clustering
# Complete
clust<-hclust(dist(cluster[,2:19]))
# rect.hclust(clust,3)
# table(cutree(clust,3))
clust.3<-cutree(clust,3)
clust.4<-cutree(clust,4)
clust.5<-cutree(clust,5)
clust.6<-cutree(clust,6)
clust.7<-cutree(clust,7)
```

```
clust.3.all<-cbind(clust.3,cluster[,2:19])
clust.4.all<-cbind(clust.4,cluster[,2:19])
clust.5.all<-cbind(clust.5,cluster[,2:19])
clust.6.all<-cbind(clust.6,cluster[,2:19])
clust.7.all<-cbind(clust.7,cluster[,2:19])

#clust.small<-array(cluster[,2:19],dim=c(1000,18))

# Average
clust.avg<-hclust(dist(cluster[,2:19]),method = "average")

clust.avg.3<-cutree(clust.avg,3)
clust.avg.4<-cutree(clust.avg,4)
clust.avg.5<-cutree(clust.avg,5)
clust.avg.6<-cutree(clust.avg,6)
clust.avg.7<-cutree(clust.avg,7)

clust.avg.3.all<-cbind(clust.avg.3,cluster[,2:19])
clust.avg.4.all<-cbind(clust.avg.4,cluster[,2:19])
clust.avg.5.all<-cbind(clust.avg.5,cluster[,2:19])
clust.avg.6.all<-cbind(clust.avg.6,cluster[,2:19])
clust.avg.7.all<-cbind(clust.avg.7,cluster[,2:19])

# Divisive
clust.divis<-diana(dist(cluster[,2:19]))

clust.divis.3<-cutree(clust.divis,3)
clust.divis.4<-cutree(clust.divis,4)
clust.divis.5<-cutree(clust.divis,5)
clust.divis.6<-cutree(clust.divis,6)
clust.divis.7<-cutree(clust.divis,7)

clust.divis.3.all<-cbind(clust.divis.3,cluster[,2:19])
```

```
clust.divis.4.all<-cbind(clust.divis.4,cluster[,2:19])
clust.divis.5.all<-cbind(clust.divis.5,cluster[,2:19])
clust.divis.6.all<-cbind(clust.divis.6,cluster[,2:19])
clust.divis.7.all<-cbind(clust.divis.7,cluster[,2:19])

# Classification analysis
library(class)

classify<-function(clust,func="linear",numclust,k.nbr=1){
  classy<-factor(levels = as.factor(1:numclust))
  names(clust)[1]<-"resp"
  if(func=="linear"){
    for( w in 1:nrow(clust)){
    class<-lda(resp~.,data = clust[-w,])
    classy[w]<-predict(class,newdata = clust[w,])$class
    }
  }
  if(func=="quadratic"){
    for( w in 1:nrow(clust)){
    class<-qda(resp~.,data = clust[-w,])
    classy[w]<-predict(class,newdata = clust[w,])$class
    }
  }

  if(func=="knn"){
    for( w in 1:nrow(clust)){
      classy[w]<-knn(train = clust[-w,-1],test=clust[w,-1],cl=clust[-w,1],k=k.nbr)
    }
  }

  AER<-1-(sum(diag(table(clust[,1],classy)))/nrow(clust))
  list(AER,classy)
}
```

```
# 3 clusters
# Complete linkage
classify.linear.3.comp<-classify(clust.3.all,numclust = 3)
classify.quad.3.comp<-classify(clust.3.all,"quadratic",numclust = 3)
classify.knn.3.comp.32<-classify(clust.3.all,"knn",numclust = 3,k=32)

# Average linkage
classify.linear.3.avg<-classify(clust.avg.3.all,numclust = 3)
classify.quad.3.avg<-classify(clust.avg.3.all,"quadratic",numclust = 3)

# Single linkage
classify.linear.3.single<-classify(clust.single.3.all,numclust = 3)
classify.quad.3.single<-classify(clust.single.3.all,"quadratic",numclust = 3)
classify.knn.3.single<-classify(clust.single.3.all,"knn",numclust = 3)
classify.knn.3.single.2<-classify(clust.single.3.all,"knn",numclust = 3,k=2)
classify.knn.3.single.5<-classify(clust.single.3.all,"knn",numclust = 3,k=5)

# Divisive
classify.linear.3.divis<-classify(clust.divis.3.all,numclust = 3)
classify.quad.3.divis<-classify(clust.divis.3.all,"quadratic",numclust = 3)
classify.knn.3.divis.32<-classify(clust.divis.3.all,"knn",numclust = 3,k=32)

# Ward's linkage
classify.linear.3.ward<-classify(clust.ward.3.all,numclust = 3)
classify.knn.3.ward.32<-classify(clust.ward.3.all,"knn",numclust = 3,k=32)

# Best clusters: divisive with 3 clusters
# Calculate E and H
clust.divis<-diana(dist(cluster[,2:19]))
clust.divis.3<-cutree(clust.divis,3)
clust.divis.3.all<-cbind(clust.divis.3,cluster[,2:19])
```

```
H<-matrix(0,ncol = 18,nrow = 18)
clust.3.means<-colMeans(clust.divis.3.all[,2:19])
clust.3.means.groups<-matrix(NA,ncol=18,nrow = 3)
for(j in 1:3){
  clust.3.means.groups[j,]<-colMeans(clust.divis.3.all[clust.divis.3.all$clust.divis.3==j,2:19])
}

n.c<-as.numeric(table(clust.divis.3.all$clust.divis.3))
for(i in 1:3){
  H<-H+n.c[i]*(clust.3.means.groups[i,]-clust.3.means)%*%t(clust.3.means.groups[i,]-clust.3.means)
}

E<-matrix(0,ncol = 18,nrow = 18)
for(i in 1:3){
  clust.3.group<-as.matrix(clust.divis.3.all[clust.divis.3.all$clust.divis.3==i,2:19])
  for(k in 1:n.c[i]){
    E<-E+(clust.3.group[k,]-clust.3.means.groups[i,])%*%t(clust.3.group[k,]-clust.3.means.groups[i,])
  }
}

# Eigenvalues/eigenvectors of invEH
eigens<-eigen(solve(E)%*%H)
eval<-zapsmall(as.numeric(eigens[[1]]))
evec<-matrix(as.numeric(eigens[[2]]),ncol = 18)

cumsum(eval)/sum(eval)

# Get the discriminant function
nu.e<-1000-3
a.star<-diag(sqrt((1/nu.e)*diag(E)))%*%evec[,1:3]
row.names(a.star)<-names(clust.divis.3.all[,2:19])
a.star[abs(a.star[,1])>max(abs(a.star[,1]))/2,1]
```

```
#=== Elbow
k.max <- 7
wss <- sapply(1:k.max,function(k){kmeans(data_pt1, k, nstart=10 )$tot.withinss})
plot(1:k.max, wss, type="b", pch = 19, frame = FALSE, xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares", main="Elbow Method")
abline(v = 3, lty =2)
#=== end Elbow
#=== Gap method
gap_stat <- clusGap(data_pt1, FUN = kmeans, nstart = 20, K.max = 7, B = 20)
plot(gap_stat, frame = FALSE, xlab = "Number of clusters k",main="Gap Method")
abline(v = 3, lty = 2)
#=== end Gap
```