Superhero or Supervillain?

Pedro Bonnin

Introduction

Can super power and demographic information for superheroes predict if they align good or bad?

About the data:

The data set chosen contains demographic and power information scrapped from https://www.superherodb.com/ and posted on kaggle (https://www.kaggle.com/claudiodavi/superhero-set). It consisted of two csv files:

- super_hero_powers.csv

```
superPowers <- read.csv("C:/Users/pbonnin/Desktop/Local
DataScience@Syracuse/IST565 Project/super_hero_powers.csv",stringsAsFactors =
FALSE)

#a data set with 667 examples with hero names and 167 different super powers.
dim(superPowers)

## [1] 667 168

#data frame is a sort of sparse matrix with character strings for "true" and
"false"
str(superPowers[,1:10])

## 'data.frame':    667 obs. of  10 variables:
##  $ hero_names           : chr  "3-D Man" "A-Bomb" "Abe Sapien" "Abin Sur"
...
##  $ Agility              : chr  "True" "False" "True" "False" ...
##  $ Accelerated.Healing  : chr  "False" "True" "True" "False" ...
##  $ Lantern.Power.Ring   : chr  "False" "False" "False" "True" ...
##  $ Dimensional.Awareness: chr  "False" "False" "False" "False" ...
##  $ Cold.Resistance      : chr  "False" "False" "True" "False" ...
##  $ Durability           : chr  "False" "True" "True" "False" ...
##  $ Stealth              : chr  "False" "False" "False" "False" ...
##  $ Energy.Absorption    : chr  "False" "False" "False" "False" ...
##  $ Flight               : chr  "False" "False" "False" "False" ...
```

- heroes_information.csv

```
superDemo <- read.csv("C:/Users/pbonnin/Desktop/Local
DataScience@Syracuse/IST565 Project/heroes_information.csv",stringsAsFactors
= FALSE)

#has 734 examples of 10 variables
str(superDemo)

## 'data.frame':    734 obs. of  11 variables:
##  $ X         : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ name      : chr  "A-Bomb" "Abe Sapien" "Abin Sur" "Abomination" ...
##  $ Gender    : chr  "Male" "Male" "Male" "Male" ...
##  $ Eye.color : chr  "yellow" "blue" "blue" "green" ...
##  $ Race      : chr  "Human" "Icthyo Sapien" "Ungaran" "Human / Radiation"
...
##  $ Hair.color: chr  "No Hair" "No Hair" "No Hair" "No Hair" ...
##  $ Height    : num  203 191 185 203 -99 193 -99 185 173 178 ...
##  $ Publisher : chr  "Marvel Comics" "Dark Horse Comics" "DC Comics"
"Marvel Comics" ...
##  $ Skin.color: chr  "-" "blue" "red" "-" ...
##  $ Alignment : chr  "good" "good" "good" "bad" ...
##  $ Weight    : num  441 65 90 441 -99 122 -99 88 61 81 ...
```

At first glance it seemed as if the data would need minimal pre-processing but there were some decisions to be made as Na's were included as "-" character strings and "-99" values. Taking out every incomplete variable would leave me with less than 100 examples in the superdemo data frame so I decided to proceed as is.

```
superDemoC <- superDemo
superDemoC[superDemoC=="-"] <- NA
superDemoC[superDemoC=="-99"] <- NA
dim(superDemoC[complete.cases(superDemoC),])

## [1] 50 11
```

Data mining problem and methodology

The goal of the project is:

- To use AR mining, thinking of each hero as a "basket" of superpowers to see who superpowers go together and to see whether there is a pattern that can predict good or bad alignment.

- Build a classification model using Naive Bayes, SVM or Random Forest to predict the alignment of the super heroes on a hold out sample.

Pre-processing

```
#libraries
library(arules)
library(arulesViz)
library(cluster)
library(EMCluster)
library(randomForest)
library(caret)
library(e1071)
library(FSelector)
library(wordcloud)
```

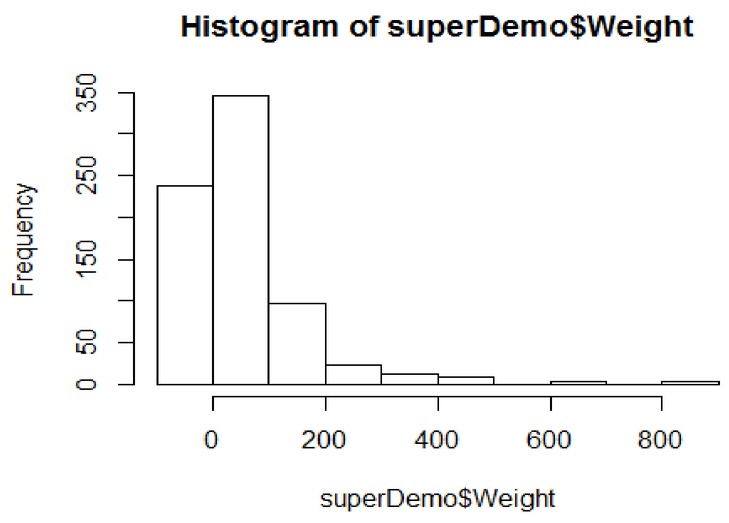
Pre-processing super powers into binary

```
SparsePowers <- superPowers
SparsePowers[SparsePowers=="True"] <- 1
SparsePowers[SparsePowers=="False"] <- 0
SparsePowers[,2:ncol(SparsePowers)] <-
as.data.frame(apply(SparsePowers[,2:ncol(SparsePowers)],2,as.numeric))
```

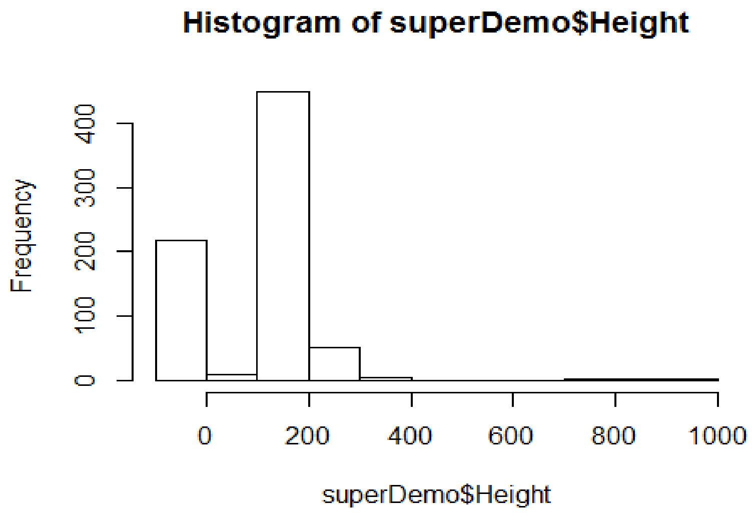
Pre-processing demographics and publisher

```
#dropping the ID number
superDemo <- superDemo[,-1]
superDemoF <- superDemo

#Looking at the weight for discretization, there are clearly a lot of
examples with "-99" these range from human sized superheroes to Godzilla so
its definitively a stand in for missing information
hist(superDemo$Weight)
```



```
#height shows the same thing as weight, "-99" values are stand ins for NA
hist(superDemo$Height)
```



```
#keeping only complete cases, turning height/weight into numeric attributes
and replacing "-" with neutral for alignment and with "flesh" for skin color.

superDemoF <- superDemoF[complete.cases(superDemoF),]
superDemoF$Height <- as.numeric(superDemoF$Height)
superDemoF$HeightNum <- superDemoF$Height
superDemoF$Weight <- as.numeric(superDemoF$Weight)
superDemoF$WeightNum <- superDemoF$Weight
superDemoF$Alignment <- as.factor(gsub("-","neutral",superDemoF$Alignment))
superDemoF$Skin.color <- as.factor(gsub("-","flesh",superDemoF$Skin.color))

#turning the rest of the character vectors into factors
superDemoF[,c("Gender","Eye.color","Race","Hair.color","Publisher")] <-
as.data.frame(apply(superDemoF[,c("Gender","Eye.color","Race","Hair.color","P
ublisher")],2,as.factor))
```

When discretizing height and weight the values were centered on what a human could possibly have. This will make it easier to understand the bins.

```
#discretizing weight and height
superDemoF$Weight <- cut(superDemoF$Weight, breaks = c(-Inf,-
90,50,200,400,Inf), labels = c("-
","Ultra.Light","Normal","Heavy","Ultra.Heavy"))
superDemoF$Weight <- ordered(superDemoF$Weight,c("-
","Ultra.Light","Normal","Heavy","Ultra.Heavy"))
superDemoF$Height <- cut(superDemoF$Height, breaks = c(-Inf,-
90,35,150,200,250,Inf), labels = c("-
","Tiny","Short","Normal","Tall","Huge"))
superDemoF$Height <- ordered(superDemoF$Height,c("-
","Tiny","Short","Normal","Tall","Huge"))
```

```
superDemoF[superDemoF==-99] <- 0

#merging the two dataframes
superDF <- merge.data.frame(superDemoF,superPowers, by.y = "hero_names" ,
by.x = "name", all.y = TRUE)

#a superDF with powers in binary instead of strings
superDF2 <- merge.data.frame(superDemoF,SparsePowers, by.y = "hero_names" ,
by.x = "name", all.y = TRUE)
superDF2 <- superDF2[complete.cases(superDF2),]
```

There are too many factors in some of the vectors for superDemo. Especially for race.

```
str(superDemoF)

## 'data.frame':    732 obs. of  12 variables:
##  $ name      : chr  "A-Bomb" "Abe Sapien" "Abin Sur" "Abomination" ...
##  $ Gender    : Factor w/ 3 levels "-","Female","Male": 3 3 3 3 3 3 3 3 2 3
...
##  $ Eye.color : Factor w/ 23 levels "-","amber","black",..: 20 4 4 9 4 4 4
4 4 7 ...
##  $ Race      : Factor w/ 61 levels "-","Alien","Alpha",..: 24 33 55 32 12
24 1 24 1 24 ...
##  $ Hair.color: Factor w/ 30 levels "-","Auburn","black",..: 18 18 18 18 4
18 7 7 7 9 ...
##  $ Height    : Ord.factor w/ 6 levels "-"<"Tiny"<"Short"<..: 5 4 4 5 1 4 1
4 4 4 ...
##  $ Publisher : Factor w/ 25 levels "","ABC Studios",..: 13 3 4 13 13 13 15
4 13 13 ...
##  $ Skin.color: Factor w/ 17 levels "black","blue",..: 4 2 13 4 4 4 4 4 4 4
...
##  $ Alignment : Factor w/ 3 levels "bad","good","neutral": 2 2 2 1 1 1 2 2
2 2 ...
##  $ Weight    : Ord.factor w/ 5 levels "-"<"Ultra.Light"<..: 5 3 3 5 1 3 1
3 3 3 ...
##  $ HeightNum : num  203 191 185 203 0 193 0 185 173 178 ...
##  $ WeightNum : num  441 65 90 441 0 122 0 88 61 81 ...
```

Need to boil race down to less factors but it is not immediately obvious how to group them so k means clustering was used to find these groups.

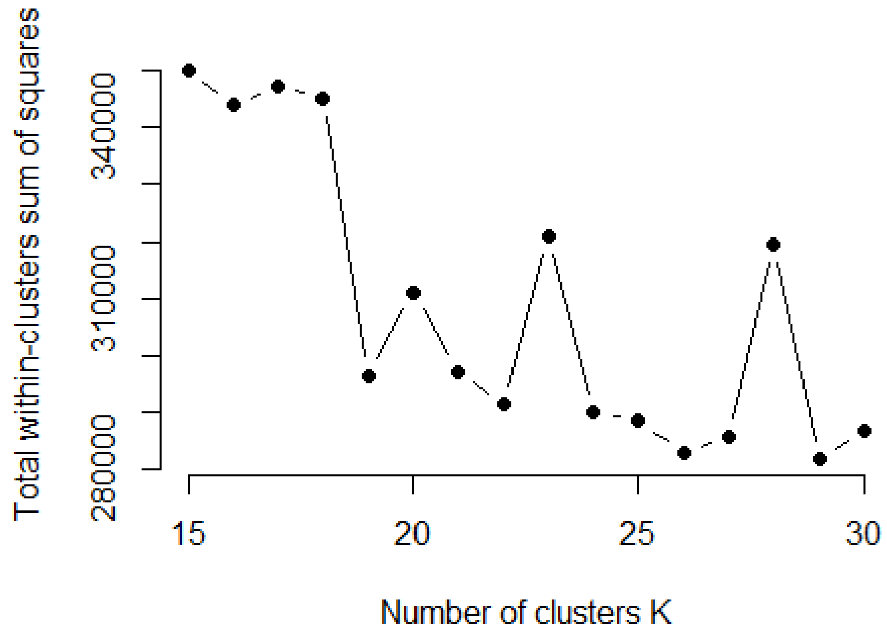
Clustering to replace Race

K means clustering was used using base R and the numeric variables (super power sparse matrix and numeric height and weight)

```
#trying out different k's using the numeric variables (super power sparse
matrix and numeric height and weight)
data <- superDF2[,11:ncol(superDF)]
```

Sum of squares was used to determine how many K's to use:

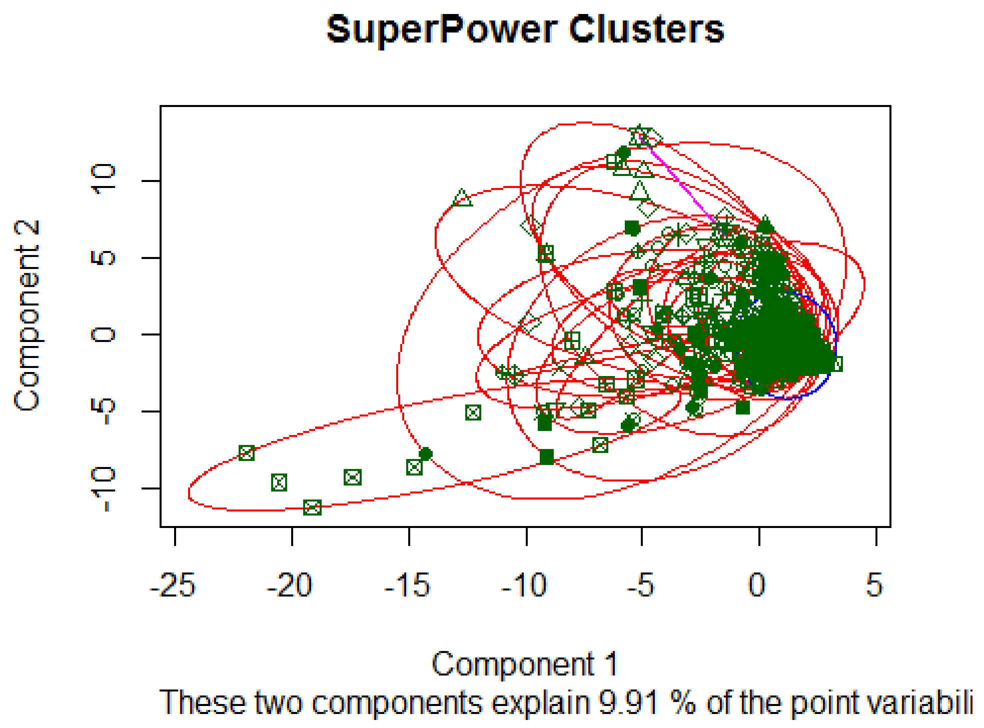
```
set.seed(50)
wss <- sapply(15:30,function(k){kmeans(data, k, iter.max = 30
)$tot.withinss})
#elbow plot for kmeans
set.seed(50)
plot(15:30, wss,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")
```



Settled on 24 since I didn't want to boil down race too much. However, the clusters do overlap a lot:

```
set.seed(50)
superKMean <- kmeans(superDF2[,11:ncol(superDF)], 24)
```

```
clusplot(superDF2[,11:ncol(superDF)],superKMean$cluster,color = TRUE,lines =
0, main = "SuperPower Clusters")
```



```
#clusters into the data frame
superDF3 <- data.frame("Clusters"=as.factor(superKMean$cluster),superDF2)
```

AR Mining

This data set lends itself very well to AR mining because of the large number of incomplete examples and the way that the are different “baskets” of superpowers and characteristics.

```
#dropping incomplete values from before to exclude from the "baskets"
superDF2AR <- superDF2
superDF2AR[superDF2AR=="-"] <- NA
superDF2AR[superDF2AR=="flesh"] <- NA
superDFTran <- as(cbind(superDF2AR[,2:10],superDF2[,13:ncol(superDF2)]==1),
"transactions")
itemFreq <- sort(itemFrequency(superDFTran),decreasing = TRUE)

#top 10 items: males, good alignment, normal weight and height, Super
strenght and stammina all appear, at least individually, in more than 40% of
the baskets
itemFreq[1:10]

##          Gender=Male        Alignment=good           Height=Normal
##            0.7036474             0.6595745               0.6367781
##           Weight=Normal          Super.Strength Publisher=Marvel Comics
##               0.5911854               0.5471125               0.5182371
##                 Stamina              Durability             Super.Speed
##               0.4437690               0.3951368               0.3799392
##                 Agility
##               0.3677812
```

Set a very high confidence threshold since a lot of rules would tie at this level. The limit on the length of the rules is set to 10 because some characters have a lot of superpowers.

```
superRules <- apriori(superDFTran, parameter = list(supp = 0.05, conf = 1,

superRules <- sort(superRules, by = "support", decreasing = TRUE)
```

Top 5 rules for confidence = 1, sorted by support show the most common co-occurrence for super strength and normal height

```
inspect(superRules[1:5])

##     lhs                          rhs              support confidence
lift count
## [1] {Durability,
##      Stamina,
##      Super.Speed,
##      Reflexes}                => {Super.Strength} 0.11702128          1
1.827778    77
## [2] {Publisher=Marvel Comics,
##      Durability,
##      Stamina,
##      Super.Speed}             => {Super.Strength} 0.11702128          1
1.827778    77
## [3] {Race=Human,
##      Weight=Normal,
##      Agility}                 => {Height=Normal}  0.10638298          1
1.570406    70
## [4] {Eye.color=blue,
##      Publisher=DC Comics,
##      Weight=Normal}           => {Height=Normal}  0.10334347          1
1.570406    68
## [5] {Weight=Normal,
##      Durability,
##      Stamina,
##      Super.Speed,
##      Reflexes}                => {Super.Strength} 0.09422492          1
1.827778    62
```

Lowering the confidence and sorting by lift yields some more interesting rules

```
superRules <- apriori(superDFTran, parameter = list(supp = 0.05, conf = 0.9,
maxlen = 10))

superRules <- sort(superRules, by = "lift", decreasing = TRUE)
```

Top 5 rules sorted by lift show high correlation between weapon based super powers and marksmanship

```
inspect(superRules[1:5])

##     lhs                  rhs              support  confidence     lift
count
## [1] {Height=Normal,
##      Stealth,
##      Weapons.Master,
##      Stamina}         => {Marksmanship} 0.05775076  0.9500000 5.041129
38
## [2] {Alignment=good,
##      Stealth,
##      Weapons.Master,
##      Stamina}         => {Marksmanship} 0.05623100  0.9487179 5.034326
37
## [3] {Race=Human,
##      Stealth,
##      Weapons.Master,
##      Stamina}         => {Marksmanship} 0.05167173  0.9444444 5.011649
34
## [4] {Height=Normal,
##      Weight=Normal,
##      Stealth,
##      Weapons.Master,
##      Stamina}         => {Marksmanship} 0.05015198  0.9428571 5.003226
33
## [5] {Gender=Male,
##      Agility,
##      Stealth,
##      Weapons.Master,
##      Stamina}         => {Marksmanship} 0.05471125  0.9230769 4.898263
36
```

What powers coincide with good and bad alignment?

Since there are a lot more good guys than bad guys, the support had to be lowered for the bad guy rules. The confidence is set relatively high so the rules are sorted by support.

```
#good guys
superRules4 <- apriori(superDFTran, parameter = list(supp = 0.05, conf = 0.9,
maxlen = 10),appearance = list(default="lhs",rhs="Alignment=good"), control =
list(verbose = F))
superRules4 <- sort(superRules4, by = "support", decreasing = TRUE)

#bad guys
superRules5 <- apriori(superDFTran, parameter = list(supp = 0.01, conf = 0.9,
maxlen = 10),appearance = list(default="lhs",rhs="Alignment=bad"), control =
list(verbose = F))
superRules5 <- sort(superRules5, by = "lift", decreasing = TRUE)
```

Features like blond hair, blue eyes and normal height and weight co-occur with being a good guy, especially when the hero is from marvel

```
inspect(superRules4[!is.redundant(superRules4)][1:10])

##      lhs                            rhs                support  confidence
lift count
## [1]  {Hair.color=Blond,
##       Height=Normal}             => {Alignment=good} 0.09726444  0.9014085
1.366652    64
## [2]  {Hair.color=Blond,
##       Weight=Normal}             => {Alignment=good} 0.08966565  0.9076923
1.376179    59
## [3]  {Eye.color=blue,
##       Hair.color=Blond,
##       Height=Normal}             => {Alignment=good} 0.08358663  0.9166667
1.389785    55
## [4]  {Eye.color=blue,
##       Hair.color=Blond,
##       Weight=Normal}             => {Alignment=good} 0.07750760  0.9107143
1.380760    51
## [5]  {Hair.color=Blond,
##       Height=Normal,
##       Publisher=Marvel Comics} => {Alignment=good} 0.06534954  0.9148936
1.387097    43
## [6]  {Height=Normal,
##       Weight=Normal,
##       Energy.Absorption}         => {Alignment=good} 0.06079027  0.9090909
1.378299    40
## [7]  {Eye.color=blue,
##       Hair.color=Blond,
##       Publisher=Marvel Comics} => {Alignment=good} 0.05927052  0.9069767
1.375094    39
## [8]  {Race=Human,
##       Height=Normal,
##       Flight}                    => {Alignment=good} 0.05775076  0.9047619
1.371736    38
## [9]  {Hair.color=Blond,
##       Height=Normal,
##       Stamina}                   => {Alignment=good} 0.05623100  0.9024390
1.368214    37
## [10] {Eye.color=blue,
##       Agility,
##       Durability}                => {Alignment=good} 0.05623100  0.9024390
1.368214    37
```

Powers related to cold, intelligence and red eye color are most associated with being a bad guy. Since the sample of bad guys is much smaller, only 8 rules were created with these parameters.

```
inspect(superRules5[!is.redundant(superRules5)])
```

```
##     lhs                        rhs               support  confidence
lift count
## [1] {Gender=Male,
##      Eye.color=red,
##      Force.Fields}          => {Alignment=bad} 0.01215805  1.0000000
3.409326     8
## [2] {Cold.Resistance,
##      Super.Speed,
##      Natural.Weapons}       => {Alignment=bad} 0.01063830  1.0000000
3.409326     7
## [3] {Cold.Resistance,
##      Stealth,
##      Weapons.Master}        => {Alignment=bad} 0.01063830  1.0000000
3.409326     7
## [4] {Hair.color=No Hair,
##      Super.Speed,
##      Natural.Weapons}       => {Alignment=bad} 0.01063830  1.0000000
3.409326     7
## [5] {Gender=Male,
##      Eye.color=red,
##      Accelerated.Healing,
##      Reflexes}              => {Alignment=bad} 0.01215805  1.0000000
3.409326     8
## [6] {Height=Normal,
##      Intelligence,
##      Super.Strength,
##      Shapeshifting}         => {Alignment=bad} 0.01063830  1.0000000
3.409326     7
## [7] {Longevity,
##      Intelligence,
##      Shapeshifting}         => {Alignment=bad} 0.01519757  0.9090909
3.099388    10
## [8] {Cold.Resistance,
##      Natural.Weapons}       => {Alignment=bad} 0.01367781  0.9000000
3.068394     9
```

Sampling the data sets

A hold out sample was created to test the accuracy of the classification models, 30% of good and bad alignment examples were randomly selected for this purpose.

```
#separating good and bad for stratified samples
goodDF <- superDF3[superDF3$Alignment=="good",]
```

```r
badDF <- superDF3[superDF3$Alignment=="bad",]
neutralDF <- superDF3[superDF3$Alignment=="neutral",]

set.seed(10)
goodIndexes <- sample(1:nrow(superDF3[superDF3$Alignment=="good",]))
badIndexes <- sample(1:nrow(superDF3[superDF3$Alignment=="bad",]))

#training and testing good DF
#length(goodIndexes)
goodTest <- goodDF[goodIndexes[1:130],]
goodTrain <- goodDF[goodIndexes[131:length(goodIndexes)],]

#training and testing bad DF
#length(badIndexes)
badTest <- badDF[badIndexes[1:60],]
badTrain <- badDF[badIndexes[61:length(badIndexes)],]

#Merging back together both samples
superTest <- rbind(goodTest,badTest)
superTrain <- rbind(goodTrain,badTrain)

superTrain$Alignment <- factor(superTrain$Alignment)
superTest$Alignment <- factor(superTest$Alignment)

#dropping variables with too many factors (using clusters instead) = names,
race, eye
superTest <- superTest[,-c(2,5,12,13)]
superTrain <- superTrain[,-c(2,5,12,13)]
superDF4 <- superDF3[,-c(2,5,12,13)]
```

Some algorithms work much better when all variables are factors so a copy of the data sets was turned into factors

```r
superTrainF <- superTrain
superTrainF$REF <- rep("Train",nrow(superTrainF))
superTrainF[,10:ncol(superTrainF)] <-
as.data.frame(apply(superTrainF[,10:ncol(superTrainF)],2,as.factor))

superTestF <- superTest
superTestF$REF <- rep("Test",nrow(superTestF))

factorizer <- rbind(superTrainF,superTestF)

factorizer[,10:ncol(factorizer)] <-
as.data.frame(apply(factorizer[,10:ncol(factorizer)],2,as.factor))

superTestF <- factorizer[factorizer$REF=="Test",]
superTestF <- subset(superTestF, select=-c(REF))
```

```
superTrainF <- factorizer[factorizer$REF=="Train",]
superTrainF <- subset(superTrainF, select=-c(REF))
```

Training the different algorithms

Random Forest, radial SVM and Naive Bayes models were trained on the sampled data and used on the hold out sample to check for accuracy

```
#Random forests
set.seed(100)
superRF_F <- randomForest(Alignment~.,superTrainF)
importance <- as.data.frame(superRF_F$importance)
superPredict <- predict(superRF_F,superTestF[,-8])

#Naive Bayes
set.seed(100)
superNB_DefaultR <- naiveBayes(Alignment~.,superTrainF)
BayesNB_Test <- predict(superNB_DefaultR,superTestF[,-8])

#SVM
set.seed(50)
superRadialSVM <- svm(Alignment~.,superTrain, kernel = "radial", cost = 500
,na.action = na.omit)
superSVM_TEST <- predict(superRadialSVM,superTest[,-8])
```

All the algorithms have a really hard time at classifying bad guys.

On overall accuracy Random Forests and Naïve Bayes performed similarly. Random forest, however, had very high precision for good guys and high recall for bad guys so if a classification model had to be built, it seems like Random Forest would be the best one.

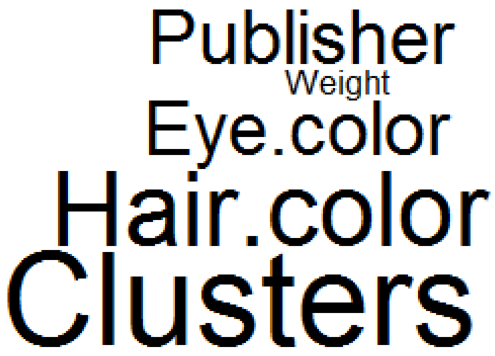
```
rbind(Bayes_Results,RF_Results,SVM_Results)

##                Accuracy Precision (good) Recall (good) Precision (bad)
## Naive Bayes   0.6842105        0.8538462     0.7302632       0.3166667
## Random Forest 0.6983240        0.9836066     0.6976744       0.0877193
## Radial SVM    0.6368421        0.7692308     0.7194245       0.3500000
##               Recall (bad)
## Naive Bayes      0.5000000
## Random Forest    0.7142857
## Radial SVM       0.4117647
```

Variable importance

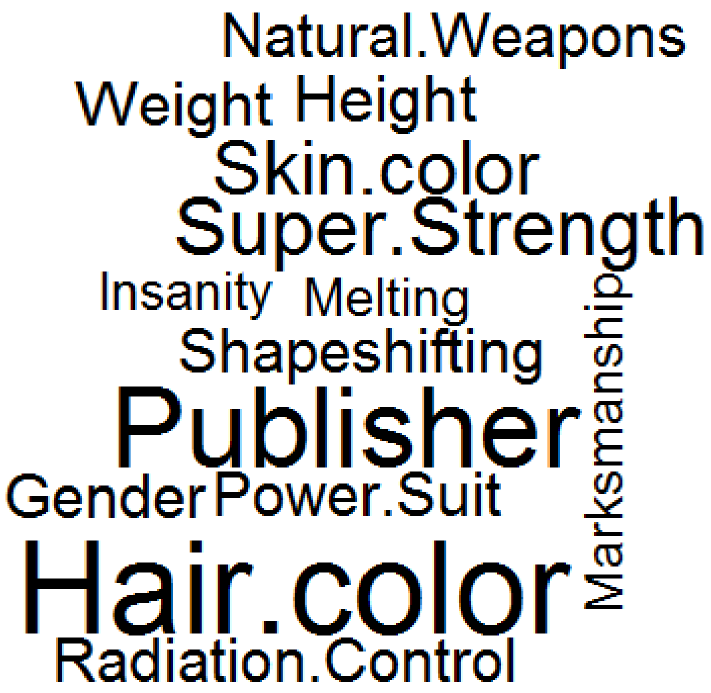
The top feature for random forest was the k-mean clusters, the rest concentrated around the "demographic" variables like hair color, publisher and eye color. Confirming some of the trends observed in the association rule results.

```
wordcloud(rownames(importance),importance$MeanDecreaseGini)
```



CHI squared variable importance also shows the clusters, hair color and eye color as the most important variables but there is not as much separation between these and some of the super powers when compared to Random Forests.

```
CHI2 <- chi.squared(Alignment~.,superTrainF)
CHI2 <- CHI2[order(-CHI2$attr_importance), , drop = FALSE]
wordcloud(rownames(CHI2),CHI2$attr_importance,colors = TRUE,max.words=20)
```



Conclusion and Discussion

The most meaningful results for this problem were derived from APRIORI, although the classification algorithms confirmed some of what was observed in the association rules.

Blond hair color, blue eyes and "human-range" weight and height seem to be good predictors of super heroes while cold related powers, intelligence and red eye color are correlated with bad guys.

For classification, all three algorithms had similar accuracy but all seemingly struggled to classify bad guys showing really low precision. This could be due to the relatively small sample of bad guys or the number of incomplete examples.

Although lacking direct real world impact, this exercise shows the skills needed in setting up and applying association rules mining, k-means clustering and classification algorithms to a novel data set.