Marcus Crowder

STA-3920

Professor Lawrence Tatum

Unsupervised Learning

With K-Means Clustering

Table Of Contents:

PartI-Introduction…3-4

PartII-Behind The Scenes...4-6

PartII-Application…7-14

PartIV-Final Thoughts…16

Part1:

<u>Intro:</u>

First I would like to give a brief into intro my topic from the last chapter of the ISRL textbook. My topic is unsupervised learning. It is a topic I am deeply interested in with the modern world of technology. These types of studies are responsible for such applications as recommendation engines and handwriting scanners that postal services use. How is this accomplished? The process involves dividing input of your data into subgroups. For example regarding a group of shoppers for a recommendation system. The data for each shopper would be analyzed based on their preferences then the algorithm finds similar buyers (or buyers with the least variance between them) and what items they have bought and recommends items to users based on that subgroup. This sounds awesome and it is even better when visualized. Similarly with automatic handwriting scanners, let's assume a person is just writing numbers the algorithm takes in a large set of numbers from 1-9 then put's each number into their own subgroup. These subgroups are placed on a plane based on specific variations found by the algorithm. When the machine has to analyze a new number input it places that new number on the plane and chooses the closest subgroup for that number. Because we know all handwritings are not the same and some are awful we would need to first have a large set of input numbers to train the data.

Issues:

The problems lying with unsupervised learning vs supervised learning is the way we test our data for accuracy. In supervised learning, we have N observations and p features we use those observations to make a prediction on the data (Y) but in unsupervised learning we are just looking for groupings of the data there is no Y response Variable. Unsupervised learning is used for exploring our data to see if there can be any possible subgroups to place our data into. We can however Label our subgroups and test the accuracy of our algorithm by hand checking and other means but these are more subjective techniques that requires a lot of hand holding with the alogirthm.

Part II

Behind The Scenes:

Principal Components Analysis:

Before diving into some R code we must talk about what is going on behind the scenes with unsupervised learning. In unsupervised learning Principal Components Analysis seeks out a specific number of variations between the different number of inputs we provide. It does not seek out all the variations just the ones that can give us a substantial amount of information from our data set. If it gave every variation the task becomes too hard to handle on large data sets, also in the wrong run variations become smaller and smaller making them less useful. For instance again with the numbers example, what is the difference between a 8 and a 9? (Think about it)

Here we see the beautiful images of an eight and a nine. The differences between these two numbers lie within the way the curves are formed. For instance the lower left of the 8 is joined to the top half while the 9 does not have that join. If Imagines of 9s and 8s were used in the PCA algorithm it would load all of the differences between the letters such as the slight curvatures , but it would put more weight on the way the 8 joins on the lower left corner and the 9 doesn't this is a distinct difference.

In our case however we will be looking at the variations in the features of our data sets using the cats and dogs example, we would use the whisker and claw length and assign weights to those variables based on their variances then plot this representation on a low-dimensional space.

More than one Principal Components are used to represent to data so as the algorithm runs it finds multiple Components to use but the components are uncorrelated this is accomplished by changing the signs of loading vectors.

FROM ISRL:

"The first principal component of a set of features X1, X1, ….,Xp is the normalized linear combination of the features that have the largest variance.

$Z_1 = \varphi_{11}X_1 + \varphi_{21}X_2 + ... + \varphi_{p1}X_p$

Principal Components use loading vectors to transform the data .

Part III

<u>Application:</u>

Now I will apply unsupervised learning practices to the Hospital Data set that was provided by the professor during the earlier parts of the semester but first I sorted the Hospitals Data in excel so the Categories go in order 25 for Mccain then 25 Straight for Obama. This is done so we can test our application better because Unsupervised learning is based on groupings and not the Response Y.

First we need to check the means and variances of our data and observe if they should be scaled or not since we don't want to have one variable outweighing another just based on a large variance.

	State	Phys	Beds	MedChg	Medicare	SocSec	SocChg	SupSec	SocEnr	Vote
2	AL	233	339	9.6	16481.06	19824.64	9.42	3595.54	903569	McCain
3	AK	240	217	24.2	7862.3	9770.5	19.35	1667.12	64843	McCain
4	AZ	244	195	16.4	13235.25	15539.43	15.96	1648.92	922932	McCain
5	AR	226	348	7.1	16924.72	20373.79	8.14	3273.23	566219	McCain
6	GA	243	277	12.3	11336.38	13593.03	12.01	2243.63	1233238	McCain
7	ID	193	246	15.4	13329.69	15831.69	15.61	1557.7	226250	McCain
8	KS	251	379	2.8	14618.9	16472.52	2.63	1426.83	452119	McCain
9	KY	253	369	8.6	16213.33	19112.93	7.85	4311.95	797660	McCain
10	LA	288	382	6.3	14130.8	15808.7	0.49	3375.56	715127	McCain
11	MS	202	453	8.9	15600.32	18807.24	6.82	4264.99	549376	McCain
12	MO	261	332	5.8	15699.82	18329.61	5.43	2030.24	1063174	McCain
13	MT	262	468	8	15778.43	18102	7.58	1581.01	169375	McCain
14	NE	267	420	3	14858.31	16601.21	2.25	1269.68	291980	McCain
15	ND	270	561	0.6	16300.42	18017.3	0.07	1241.92	114712	McCain
16	OK	194	307	6.2	15205.15	17915.44	6.98	2247.62	635619	McCain
17	SC	256	267	13.6	15154.89	18295.3	13.06	2480.63	778480	McCain
18	SD	247	598	4.3	16056.8	18142.42	3.34	1616.38	140773	McCain
19	TN	286	346	10.4	15354.94	18273.63	9.95	2705.4	1089649	McCain
20	TX	232	259	11.4	11034.87	12914.41	12.06	2205.09	2952230	McCain
21	UT	233	187	14.1	9517.35	11056.31	13.26	915.38	273045	McCain
22	WV	255	409	5.2	19564.9	22789.53	5.2	4228.18	414053	McCain
23	WY	216	405	8.5	13941.85	16001.56	7.07	1136.08	81495	McCain
24	CA	295	201	8.2	11683.97	12354.3	6.06	3348.38	4463873	Obama
25	CO	292	201	11.2	11139.94	12530.2	9.43	1190.35	584556	Obama
26	CT	401	224	2.3	15014.63	16670.93	1.25	1488.76	585199	Obama
27	DE	280	236	13.1	15030.87	17794.51	13.4	1632.08	150101	Obama
28	FL	293	287	8.5	17107.28	19281.79	7.34	2378.93	3430205	Obama
29	HI	351	250	10.2	14283.55	15742.15	9.22	1784.36	200743	Obama
30	IL	298	274	3.7	13278.78	14831.94	2.87	2026.38	1893055	Obama

```
> Hospital3 = read.csv('~/Downloads/Hospital (2).csv')
> head(Hospital3)
  State Phys Beds MedChg Medicare   SocSec SocChg  SupSec  SocEnr   Vote
1    AL  233  339    9.6 16481.06 19824.64   9.42 3595.54  903569 McCain
2    AK  240  217   24.2  7862.30  9770.50  19.35 1667.12   64843 McCain
3    AZ  244  195   16.4 13235.25 15539.43  15.96 1648.92  922932 McCain
4    AR  226  348    7.1 16924.72 20373.79   8.14 3273.23  566219 McCain
5    GA  243  277   12.3 11336.38 13593.03  12.01 2243.63 1233238 McCain
6    ID  193  246   15.4 13329.69 15831.69  15.61 1557.70  226250 McCain
> Hospital = Hospital3[,-1]
> head(Hospital)
  Phys Beds MedChg Medicare   SocSec SocChg  SupSec  SocEnr   Vote
1  233  339    9.6 16481.06 19824.64   9.42 3595.54  903569 McCain
2  240  217   24.2  7862.30  9770.50  19.35 1667.12   64843 McCain
3  244  195   16.4 13235.25 15539.43  15.96 1648.92  922932 McCain
4  226  348    7.1 16924.72 20373.79   8.14 3273.23  566219 McCain
5  243  277   12.3 11336.38 13593.03  12.01 2243.63 1233238 McCain
6  193  246   15.4 13329.69 15831.69  15.61 1557.70  226250 McCain
> apply(Hospital, 2, mean)
    Phys     Beds   MedChg Medicare   SocSec   SocChg   SupSec   SocEnr     Vote
      NA       NA       NA       NA       NA       NA       NA       NA       NA
Warning messages:
1: In mean.default(newX[, i], ...) :
  argument is not numeric or logical: returning NA
2: In mean.default(newX[, i], ...) :
  argument is not numeric or logical: returning NA
3: In mean.default(newX[, i], ...) :
```

Using the apply function I encountered my first error. The Vote column was not made of variables but Strings of McCain or Obama so we have to split out the column. (I also removed the state column because it would not be needed as I changed it to row variables)

As we can see the means have vastly different means that are different so we may need to scale them. Additionally looking at the variances we see that they are extremely small which may cause problems in clustering.

```
> apply(Hospital[,-9], 2, mean)
       Phys        Beds      MedChg    Medicare      SocSec      SocChg      SupSec      SocEnr
   285.5000    296.3200      8.3480  14482.0480  16660.3514      7.5774   2150.0764 943310.4200
> apply(Hospital[,-9], 2, var)
        Phys         Beds       MedChg     Medicare       SocSec       SocChg       SupSec       SocEnr
4.209929e+03 8.897569e+03 2.673847e+01 4.372555e+06 6.024782e+06 2.410040e+01 7.147877e+05 8.805126e+11
```

```
> row.names(Hospital) <-Hospital3[,1]
> head(Hospital)
   Phys Beds MedChg Medicare   SocSec SocChg  SupSec  SocEnr   Vote
AL  233  339    9.6 16481.06 19824.64   9.42 3595.54  903569 McCain
AK  240  217   24.2  7862.30  9770.50  19.35 1667.12   64843 McCain
AZ  244  195   16.4 13235.25 15539.43  15.96 1648.92  922932 McCain
AR  226  348    7.1 16924.72 20373.79   8.14 3273.23  566219 McCain
GA  243  277   12.3 11336.38 13593.03  12.01 2243.63 1233238 McCain
ID  193  246   15.4 13329.69 15831.69  15.61 1557.70  226250 McCain
```

Next we use the prcomp() function which is the R function to find the principal Components of our Hospital Data minus column 9 (vote column)

```
> pr.out=prcomp(Hospital[,-9], scale=TRUE)
> names(pr.out)
[1] "sdev"     "rotation" "center"   "scale"    "x"
> pr.out$center
     Phys       Beds     MedChg   Medicare     SocSec     SocChg     SupSec     SocEnr
 285.5000   296.3200     8.3480 14482.0480 16660.3514     7.5774  2150.0764 943310.4200
> pr.out$scale
        Phys         Beds       MedChg     Medicare       SocSec       SocChg       SupSec       SocEnr
6.488396e+01 9.432693e+01 5.170925e+00 2.091065e+03 2.454543e+03 4.909216e+00 8.454512e+02 9.383563e+05
> pr.out$rotation
                PC1         PC2         PC3         PC4         PC5         PC6         PC7          PC8
Phys      0.11551291  0.59623975  0.11247633 -0.52979355 -0.25639977  0.51565400  0.06933484  0.037089009
Beds      0.34348904 -0.32888062  0.20003367  0.50433329 -0.33265516  0.60311508  0.07120037 -0.003244107
MedChg   -0.45363826 -0.29490604 -0.22965938 -0.17073144 -0.07595867  0.35497667 -0.67154430  0.206398159
Medicare  0.47314782 -0.20983309 -0.08412196 -0.29429893  0.32988031  0.10469556 -0.30618993 -0.651826968
SocSec    0.43432512 -0.31765788 -0.14699005 -0.32297660  0.31898323  0.05676445  0.17809310  0.669212629
SocChg   -0.43794983 -0.32615342 -0.24354370 -0.20408316  0.12239527  0.31338875  0.64062099 -0.277978517
SupSec    0.23934512 -0.04670839 -0.70587872 -0.06423015 -0.61856589 -0.22138380  0.05006401 -0.063253518
SocEnr    0.03593137  0.44257422 -0.55527587  0.44622450  0.45788022  0.28798969 -0.02793477  0.044907453
```

I Named the variable created from prcomp(Hospital[,-9, scale=TRUE) to pr.out. Additionally I decided to scale the data because of the large differences between the means.

When we use names(pr.out)

We see

> names(pr.out)

[1] "sdev"   "rotation" "center"   "scale"   "x"

For the principal components sdev, rotation center scale and x were produced.

The Sdev relates to the standard deviation of all the principal components

> pr.out$sdev

[1] 1.87362073 1.30321827 1.10464932 0.93306982 0.67143743 0.47438843 0.14124684 0.06689407

Using the

Pr.out$Center function shows us all of the initial mean values of our hospital data predictor variables.

While pr.out$scale shows us all the scaled means of our hospital data that will be used.

pr.out$scale

Phys	Beds	MedChg	Medicare	SocSec	SocChg	SupSec	SocEnr
6.488396e+01	9.432693e+01	5.170925e+00	2.091065e+03	2.454543e+03	4.909216e+00	8.454512e+02	9.383563e+05

again the variances are extremely small

pr.out$x represent our transformed data that has gone through the pr.out$rotation loading vectors

It is super neat that we don’t have to do the calculations ourselves R is such a beautiful beast.

```
> pr.out$rotation
                PC1         PC2         PC3         PC4         PC5         PC6         PC7          PC8
Phys      0.11551291  0.59623975  0.11247633 -0.52979355 -0.25639977  0.51565400  0.06933484  0.037089009
Beds      0.34348904 -0.32888062  0.20003367  0.50433329 -0.33265516  0.60311508  0.07120037 -0.003244107
MedChg   -0.45363826 -0.29490604 -0.22965938 -0.17073144 -0.07595867  0.35497667 -0.67154430  0.206398159
Medicare  0.47314782 -0.20983309 -0.08412196 -0.29429893  0.32988031  0.10469556 -0.30618993 -0.651826968
SocSec    0.43432512 -0.31765788 -0.14699005 -0.32297660  0.31898323  0.05676445  0.17809310  0.669212629
SocChg   -0.43794983 -0.32615342 -0.24354370 -0.20408316  0.12239527  0.31338875  0.64062099 -0.277978517
SupSec    0.23934512 -0.04670839 -0.70587872 -0.06423015 -0.61856589 -0.22138380  0.05006401 -0.063253518
SocEnr    0.03593137  0.44257422 -0.55527587  0.44622450  0.45788022  0.28798969 -0.02793477  0.044907453
```

> dim(pr.out$x)

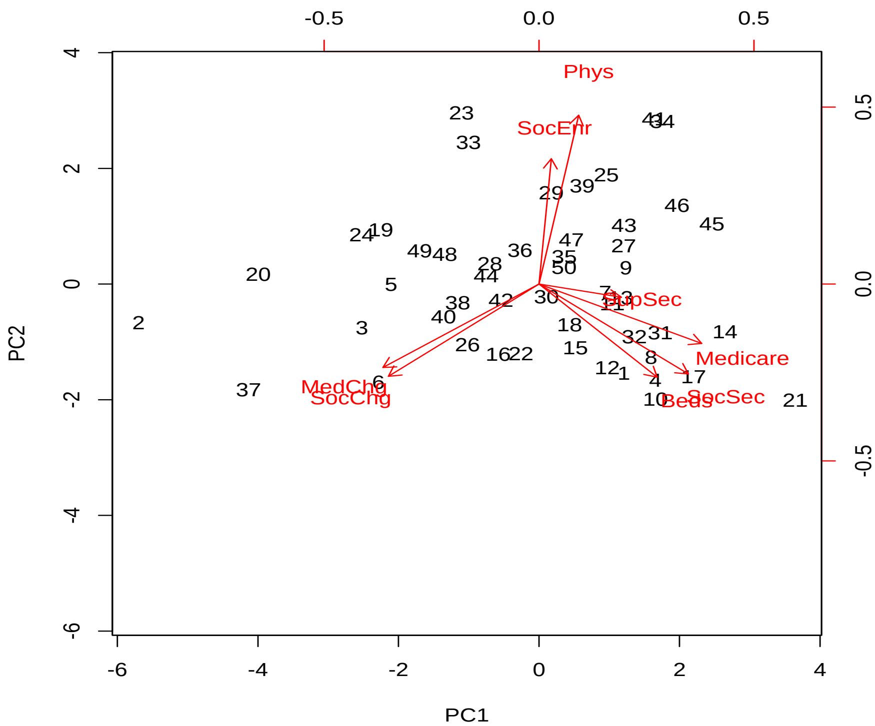
[1] 50 8

pr.out$x we see the data set is of 50 observations with 8 features which is exactly our hospital data just transformed through the loading vectors components. For example in the first Principal component Medicare is given a weight of 0.4731472 but in the second it is given a negative weight of -0.20983309 because we want the components to not be correlated with each other.

Now Lets plot our first two components with the states and the distinct weight of their features on the two Principal components.

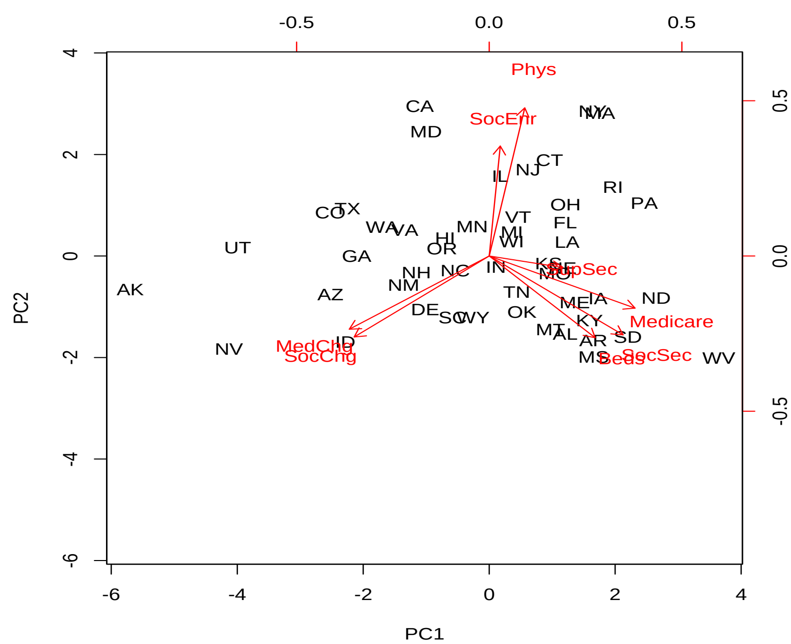
biplot(pr.out, scale = 0)

This graph shows the weights of the variables and features and their weights on our Principal components for example Phys was given a weight close to 0.1 on our first principal component but on principal component 2 it was given a weight of 3.8. We read the graph like that. But I don't like this graph I would like to see State Names instead of indexes. Thank YOU for R.




```
> row.names(Hospital3) <-Hospital3[,1]
> head(Hospital3)
   State Phys Beds MedChg Medicare   SocSec SocChg  SupSec  SocEnr   Vote
AL    AL  233  339    9.6 16481.06 19824.64   9.42 3595.54  903569 McCain
AK    AK  240  217   24.2  7862.30  9770.50  19.35 1667.12   64843 McCain
AZ    AZ  244  195   16.4 13235.25 15539.43  15.96 1648.92  922932 McCain
AR    AR  226  348    7.1 16924.72 20373.79   8.14 3273.23  566219 McCain
GA    GA  243  277   12.3 11336.38 13593.03  12.01 2243.63 1233238 McCain
ID    ID  193  246   15.4 13329.69 15831.69  15.61 1557.70  226250 McCain
```

I used the row.names(Hospital) function to swap out the names of my rows with the state names. Which is better for visualization than indexes. And this beauty was created we can now find the values of each state on our two principal components and which are

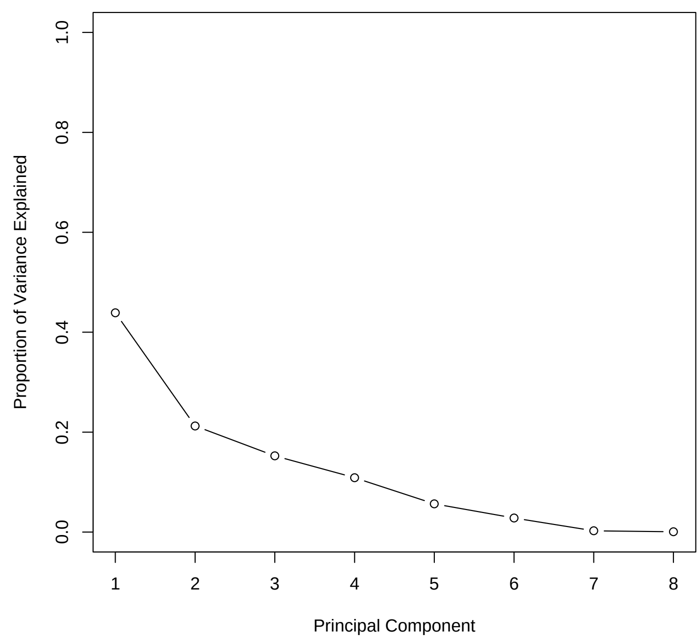


grouped together. It requires a lot of eye testing so that's why we won't do that but we will instead have K-means clustering do that work for us.

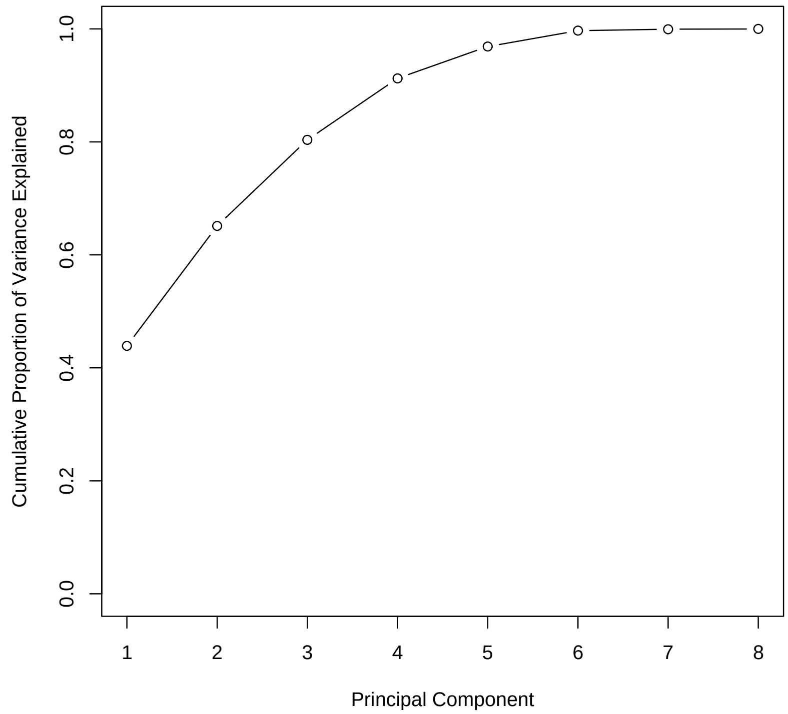
But before that earlier I did say that principal components are used to explain the variances in the data set but . it does not use all in actuality each component explains a bit of the variance so let's see which components explain the most variance and how many does it take to reach 100%

```
> pr.out$sdev
[1] 1.87362073 1.30321827 1.10464932 0.93306982 0.67143743 0.47438843 0.14124684 0.06689407
> pr.var=pr.out$sdev^2
> pr.var
[1] 3.510454651 1.698377863 1.220250118 0.870619285 0.450828218 0.225044380 0.019950669 0.004474816
> pve=pr.var/sum(pr.var)
> pve
[1] 0.4388068314 0.2122972328 0.1525312648 0.1088274106 0.0563535272 0.0281305475 0.0024938337 0.0005593521
> plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1), type='b')
> plot(cumsum(pve), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained", ylim=c(0,1), type='b')
```

To find the variance of our pr.out variable we just square the standard deviation with pr.out$sdev^2 then place it into the pve.var variable then we plot the graph . along the graph we see which principal component explains what starting with component 1 explaining .4388% of the variation then the 2nd explaining .21% and eventually it drops off at the 5th component explaining only 5%. This is why it's not always necessary to use too many components since they explain less and less in the long run causing unnecessary complexity.



Here is the same plot of the variation explained by the components but we use the cumsum(pve) function to add up the variations along the way until we reach 100%.

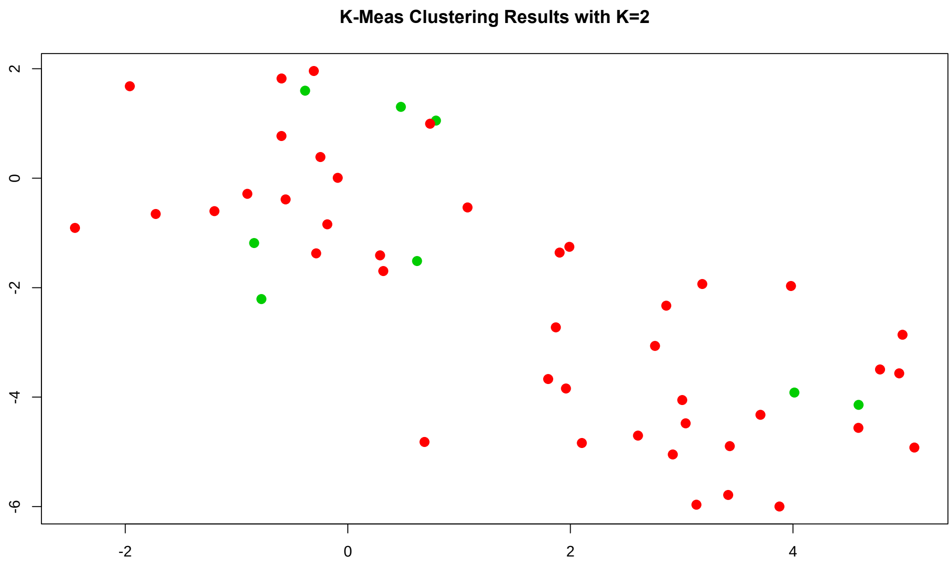


Kmeans Function:

```
> set.seed(2)
> km.out=kmeans(Hospital[,-9],2, nstart=20)
> km.out$cluster
AL AK AZ AR GA ID KS KY LA MS MO MT NE ND OK SC SD TN TX UT WV WY CA CO CT DE FL HI IL IN IA ME MD MA MI MN NV NH NJ NM NY NC OH OR PA RI VT VA WA WI
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  1  1  1  2  1  1  1  2  1  2  1  1  1  1  1  2  1  1  1  1  1  2  1  2  1  2  1  1  1  1  1
> plot(x, col=(km.out$cluster+1), main="K-Meas Clustering Results with K=2", xlab="", ylab="", pch=20, ce2)
```

first we set the seed in order to reproduce these variables in the future then we use the kmeans function to find the clusters Keams(Hospital[,-9],2,nstart=20). We use the 2 variable to say we want to find 2 clusters because our data orignall are just for Obama and Mccain.

However we are not met with favorable results for this approach as we see the data points are quit scattered with votes for McCain appearing consistently as the 1 variable but only a few 2 variables. Remember the first few 25 votes are for Mccain and the 2nd 25 are for Obama the Mccain Votes stop at WY, but there is 1 Obama decision before that and after that Mcain appear 21 times this means that we failed hard let's look at the graph of this plot.



Red dots represent McCain and Green dots represent Obama as we see the data points are grouped together but the groupings of the green dots do not make a whole lot of sense which may partly be due to the low variances as mentioned earlier. A better data set should have been used but this was fun to visualize.

<u>Extra:</u>

In most cases we do not know how many clusters are needed so before performing this exercise one must think hard about the number of clusters that could be in their data. As an example I Switched the number of clusters from 2 to 3 for my hospital data

```
> km.out
K-means clustering with 3 clusters of sizes 10, 23, 17

Cluster means:
        [,1]        [,2]
1  2.3001545 -2.69622023
2 -0.3820397 -0.08740753
3  3.7789567 -4.56200798

Clustering vector:
 [1] 3 1 3 1 3 3 3 1 3 1 3 1 3 1 3 1 3 3 3 3 3 1 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2

Within cluster sum of squares by cluster:
[1] 19.56137 52.67700 25.74089
 (between_SS / total_SS =  79.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"
```

Now we have different results where some of the variables that were in cluster 1 are now in cluster 3. Be careful with this.

PartIV:

Final Thoughts:

It was fun to explore unsupervised learning and to learn about Principal Components and how they are used but making a data set for these tasks may prove to be a big challenge for me moving on into the future. This was a great learning experience and now I know to spend maybe a few weeks on projects instead of days because it is super fun.

Wishing You All the best Prof Tatum ~~~ Foreigner Out~