Cluster Analysis and Interpreting the Results

Sai Teja Kanneganti

13 December 2016

Introduction

This document explains about several types of clustering analysis for a sample data and its interpretations. It recommends the best suitable cluster solutions for the sample data by comparing the results of several cluster analysis.

Dataset (a and b)

Dataset we are going to use exercise if **biopsy** provided by the R-Package **MASS**. **biopsy** is a breast cancer database obrained from University of Wisconsin Hospitals, Madison from Dr. William H. wolberg. He assessed bipsies of breast tumors for 699 patients. There are nine attributes in this dataset and each of the attributes has been scored on scale 1 to 10, and the outcome is also known. The dimension to the dataset is

```
## [1] 699  11
```

And following are the 9 attributes available.

- V1 - clump thickness
- V2 - uniformity of cell size
- V3 - uniformity of cell shape
- V4 - marginal adhesion
- V5 - single eithelial cell size
- V6 - bare nuclei
- V7 - bland chromatin
- V8 - normal nucleoli
- V9 - mitoses

There is an outcome column in the dataset, **"benign"** or **"malignant"**. The distribution of outcome is,

```
##
##    benign malignant
##       458       241
```

Cluster Analysis (c)

Cluster analysis is an important element of exploratory data analysis. It is typically directed to study the internal structure of a complex data set, which can not be described only through the classical second order statistics.

Clustering is an unsupervised learning method. It is a process of grouping together the data which has similar features. And each groups are called as **Clusters**. Here algorithm doesn't require any prior knowledge for grouping the data. It purely depends on the features of the data. Let us apply the following types of clustering to our dataset and interpret the results:

- Partition Clustering

- Hierarchial Clustering
- Density-based Clustering

Since we have a prior knowledge from the dataset that, it has to main classes (benign and malignant). Lets also compare how 2-cluster analysis fares with the original classes in the dataset. It should give a better idea about the accuracy of different algorithms

1. Partitional Clustering

Here we will use K-Means clustering. It is the process of grouping data in to 'K' number of clusters. For **K-Means** clustering, lets use **kmeans** function in **stats** package.

Lets start with 2 clusters,

```
bp <- biopsy[complete.cases(biopsy),]
c2 <- kmeans(bp[,2:10],2)

## ClusterSize
c2$size
```

```
## [1] 453 230
```

```
## Confusion Matrix
table(bp$class, c2$cluster)
```

```
##
##               1   2
##   benign    435   9
##   malignant  18 221
```

Similarly confusion matrix for **kmeans** with 3 clusters is,

```
##
##               1   2   3
##   benign    434  10   0
##   malignant  14 101 124
```

Similaryly, the result for **kmeans** 4,5 and 6 clusters are,

```
## [1] "kmeans with 4 Clusters"
```

```
##
##               1   2   3   4
##   benign    261   9   0 174
##   malignant   0  97 124  18
```

```
## [1] "kmeans with 5 Clusters"
```

```
##
##               1   2   3   4   5
##   benign      4   6   0 432   2
##   malignant  49  69  58   9  54
## [1] "kmeans with 6 Clusters"
```

```
##
##              1   2   3   4   5   6
##   benign      1   5   0 244  12 182
##   malignant  51  84  76   2  26   0
```

When we look at all the above results, there isn't much difference. More or less, everything gives same result in terms of confusion matrix when compared to the existing outcomes from the database. But look at the result for **6-Cluster** analysis, there is one cluster where both "benign" and "malignant" numbers are too high to combine with a single class. So it is clear, that is the cluster and those are the only data points whose features are clearly not differentiable. And size of that cluster being small, this critical insight can be used by any supervised learning algorithm to make the algorithm's performance better.

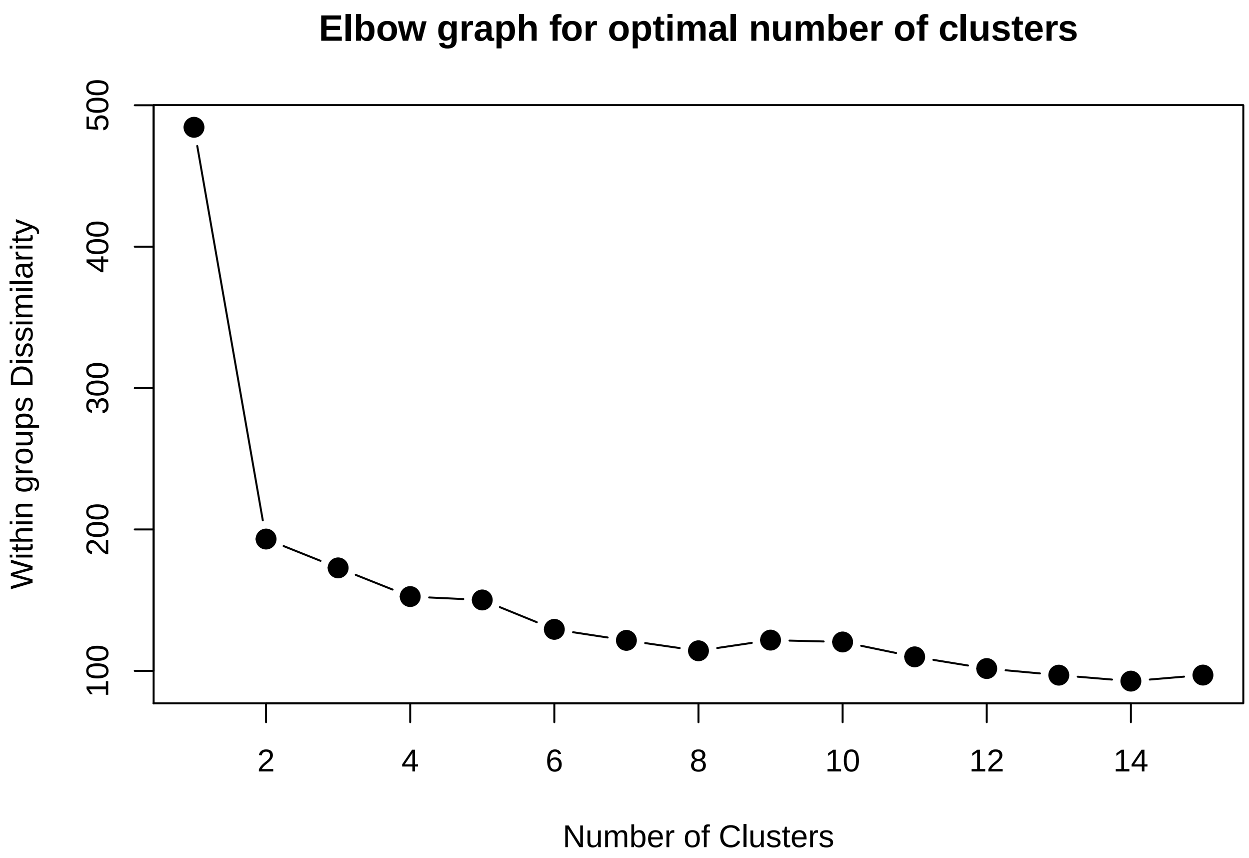
Optimal Number of Clusters

We found every clusters to be similar and 6-Cluster Analysis giving better insight than other clusters. However, one solution often used to identifiy the optimal number of clusters is called the **Elbow** method and it involves observing a set of possible numbers of clusters relative to how they minimise the within-cluster sum of squares. In other words, the Elbow method examines the within-cluster dissimilarity as a function of the number of clusters. Lets plot **Elbow** graph,

```
dt <- bp[,2:10]
dissim <- (nrow(dt)-1)*sum(apply(dt,2,var))
for (i in 2:15) dissim[i] <- sum(kmeans(dt,
                                     centers=i)$withinss)

dissim <- dissim/100

plot(1:15, dissim, type="b", xlab="Number of Clusters",
     ylab="Within groups Dissimilarity",
     main="Elbow graph for optimal number of clusters",
     pch=20, cex=2)
```



The above graph also confirms our observation. There is not much difference between the clusters from 2 to 6. But after 6-Clusters, the graph almost flat. So we can decide the **optimal number of clusters to be 2 or 3**. But since we are getting litlte more insight on the data with 6-Cluster analysis, it can be used wherever it is needed.

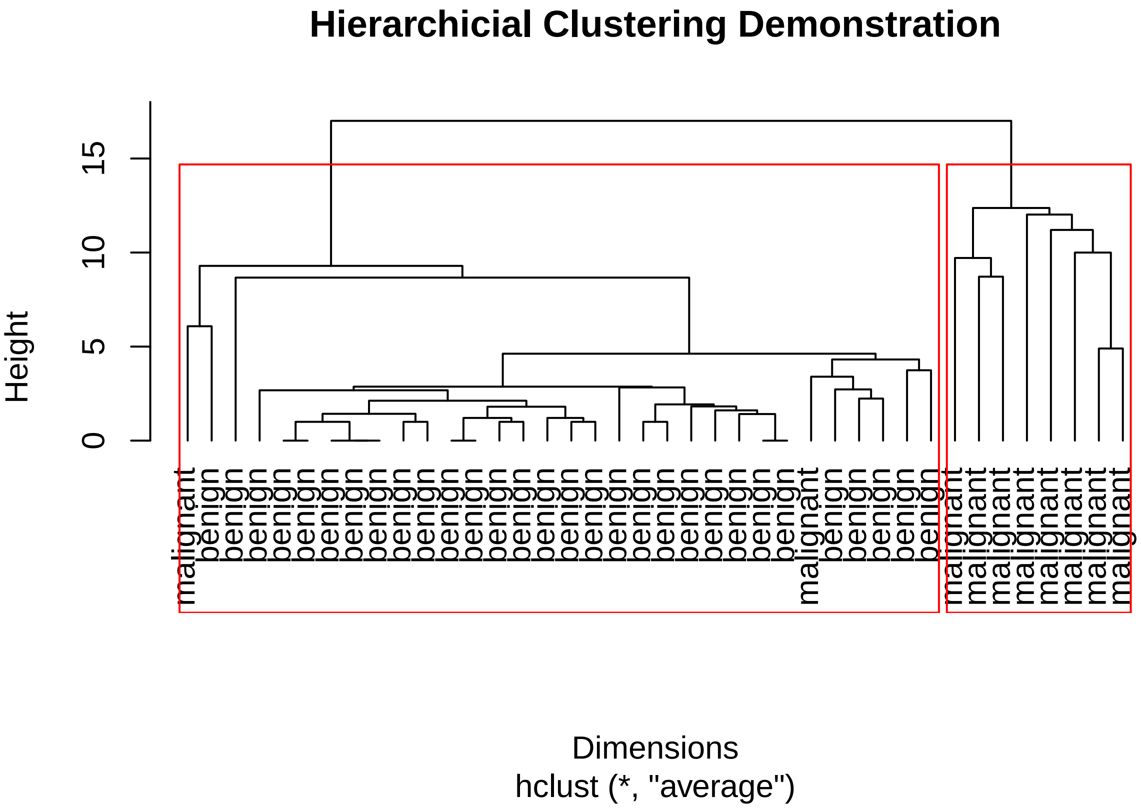
2. Hierarchical Clustering

Hierarchical Clustering is a bottom up (agglomerative) clustering approach. Two close datapoints are merged to form a single cluster and so on till it reaches single cluster. At each level of hierarchy, the number of clusters are different. The number of clusters keep reducing from bottom of the hierarchy to top of the hierarchy. **The closeness of the point is defined by the distance measure. Here it is Euclidean Distance.**

In this exercise, we will be using hclust function from the stats package for hierarchial clustering. For the purpose of visualizing hierarhcical clustering, lets do it with a small sample of data.

```
id <- sample(1:nrow(bp), 40)
bpSample <- bp[id,]
hc <- hclust(dist(bpSample[,2:10]), method="ave")

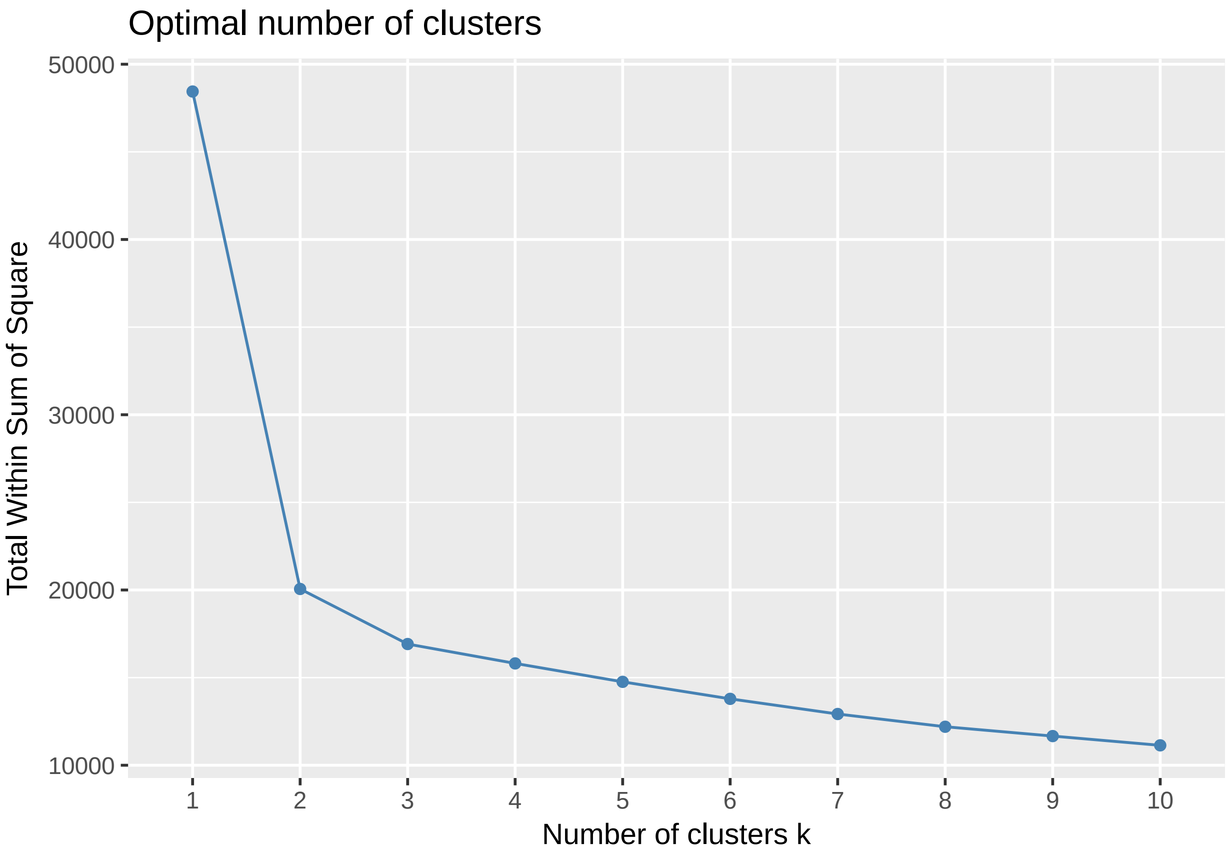
plot(hc, hang=-1, labels=bpSample$class, xlab="Dimensions", main="Hierarchicial Clustering Demonstratio
rect.hclust(hc,k=2)
```



Optimal Number of Clusters

Lets use same **Elbow method** to find the optimal number clusters in our dataset. Lets use a library called **factoextra**, which has predefined function to calculate within cluster sum of squares and plot it.

```
fviz_nbclust(bp[,2:10], hcut, method="wss" )
```



This graph is very similar to the one which we got with respect to kmeans clustering. **The optimal number of clusters is 2 or 3**. Let us calculate the confusion matrix to confirm the result we get.

```
hcFull <- hclust(dist(bp[,2:10]), method="ave")
bp$hc2 <- cutree(hcFull, k=2)
bp$hc3 <- cutree(hcFull, k=3)
table(bp$class, bp$hc2)
```

```
##
##               1   2
##   benign    436   8
##   malignant  31 208
```

```
table(bp$class, bp$hc3)
```

```
##
##               1   2   3
##   benign    436   8   0
##   malignant  31 207   1
```

From the above we can confirm that the optimal number of clusters is 2 or 3.

3. Density-based Clustering

The goal of density based clustering is to identify the dense regions in a data sapce. It is measured by number if objects close to a given point. The main advantage of using this algorithm is to avoid the noise in the data.

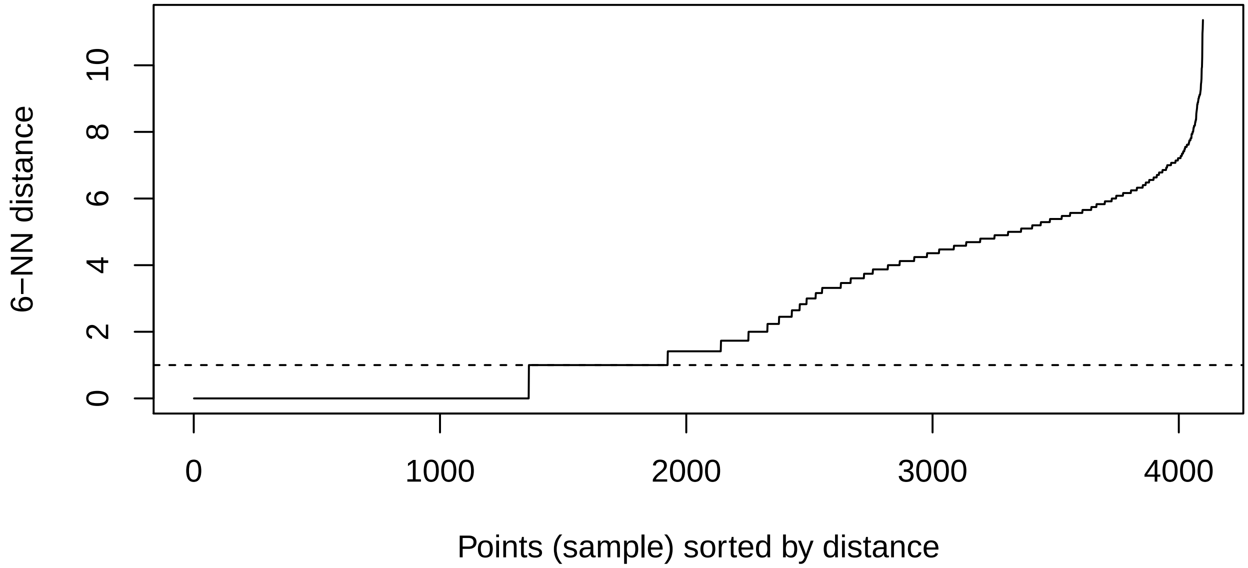
In the case of KMeans and Hierarchical clustering, each data point is clustered in a spearate group whether it is noise or not. But in this case noisy can be avoided. And Density-based clustering can find any shape of cluster.

The algorithm we are going to use is **dbscan** from the package **fpc**. There are 2 key parameters in dbscan:

- **eps**: reachability distance, which defines the size of neighbourhood,
- **MinPts**: minimum number of points

Optimal eps:

We can calcuate K-nearest neighbout distances in a matrix of point. The idea is to calculate the average of the distances of every point to its k-nearest neighbour. The value of 'k' will be specified by the users and corresponds to MinPts. The function **kNNdistplot()** from the package **dbscan** calculates these distances and plots for us. From the graph we could find optimal eps.



We re getting similar graph for whatever the value of k. If **1.0** can be used as eps value as per the above graph, we can use **0.1** as well. So, optimal eps can be 1.0 or 0.1 And following are the outcomes.

- **eps=1.0**: we are geting 2 clusters irrespective of the whatever the value of K. The confusion matrix is,

```
##
##               0   1
##   benign    125 319
##   malignant 239   0
```

- **eps=0.1**: We are getting 3 clusters if the value of MinPts=22, 23. And we are getting 2 clusters if the value of MinPts=24,26,26,27. And the confusion matrix for the 2-cluster is,

```
##
##               0   1
##   benign    417  27
##   malignant 239   0
```

From the above confusion matrices, it is evident that optimal values are **eps = 0.1** and **MinPts=22 to 27**

Cluster Description and Results (d)

1. K-Means

~~~~
* Prefered Number of Clusters: 2 or 3
* Method to derive prefered Clusters: Elbow Method
* Confusion Matrix with 2 Clusters:

    ```
    ##
    ##                1   2
    ##   benign     435   9
    ##   malignant   18 221
    ```

* Confusion Matrix with 3 Clusters:

    ```
    ##
    ##                1   2   3
    ##   benign     434  10   0
    ##   malignant   14 101 124
    ```
~~~~

2. Hierarchical Clustering

~~~~
* Prefered Number of Clusters: 2 or 3
* Method to derive prefered Clusters: Elbow Method
* Confusion Matrix with 2 Clusters:

    ```
    ##
    ##                1   2
    ##   benign     436   8
    ##   malignant   31 208
    ```
* Confusion Matrix with 3 Clusters:

    ```
    ##
    ##                1   2   3
    ##   benign     436   8   0
    ##   malignant   31 207   1
    ```
~~~~

3. Density Based Clustering

~~~~
* Prefered epsilon(eps): 0.1, Prefered MinPts: 22 to 27
* Method to derive Prefered eps: KNN Distance Plot
* Confusion Matrix with 2 Clusters:

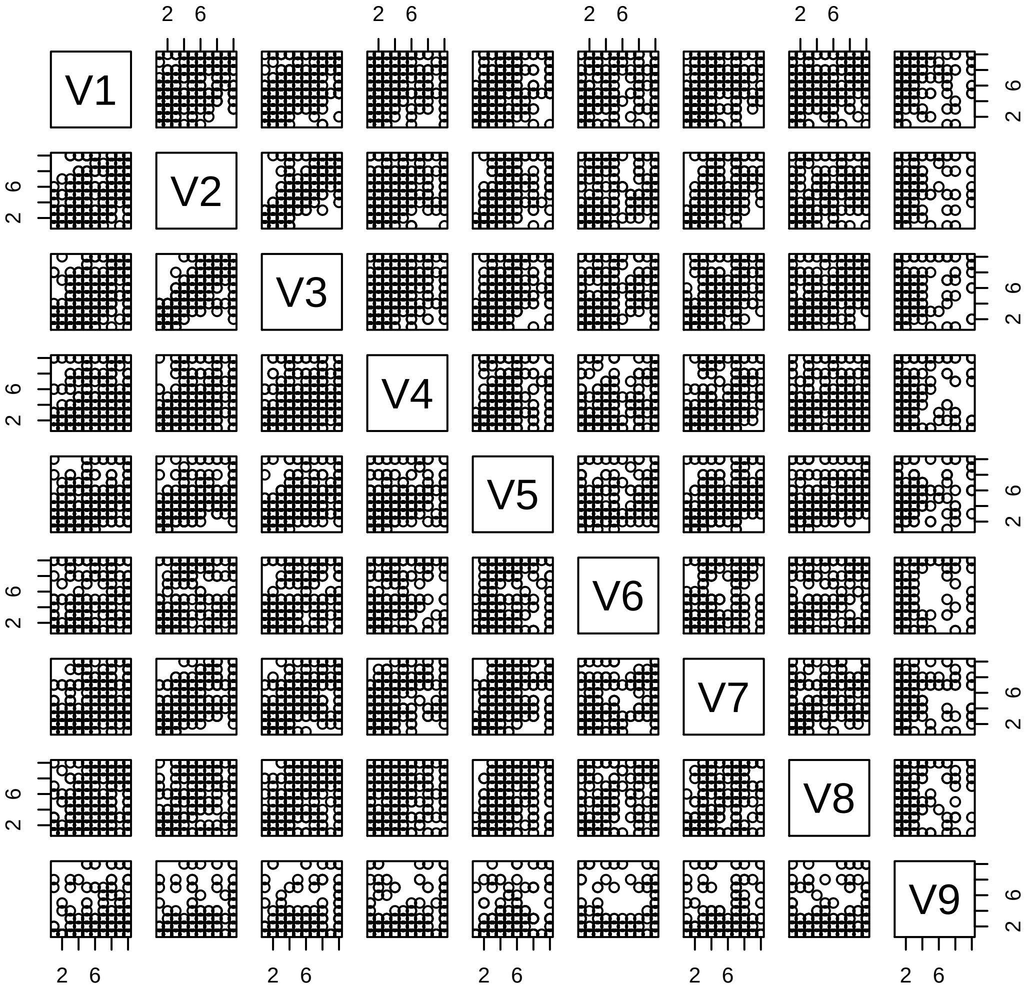
    ```
    ##
~~~~

~~~
        ##                0   1
        ##   benign     125 319
        ##   malignant 239   0
        ```
    * Confusion Matrix with 3 Clusters:

        ```
        ##
        ##                0   1
        ##   benign     417  27
        ##   malignant 239   0
        ```
~~~

Summary

Since there is no other means to compare the three types of clusterings with our dataset, we have used the existing lables and confusion matrix. We have just compared the existing lables with new clusters and how checked how it fared in each of the Algorithm. Different algorithms will be suitable for different kind of data. We can clearly see that **Density based clustering is not suitable for this data** (Check KNN distance Plot). One reason is because there is not clear separation between the data points based on Density. Since all the variables are scores between 1 to 10, density is almost same with respect to the data elements. The below plot verifies that:



Prefered Solution (e)

Prefered solutions in this exercise is **K-Means Clustering**. And following are the reasons for the preference:

- The data already had predefined classes. So, we had good idea by comparing the output with exising classes. In that way, K-Means clustering gave better Precision and Recall.
- When, increasing the number of clusters, K-Means gave some better insights as well than the other 2 algorithms. For example, in 6-Cluster classification, every other cluster had data points which corresponds to one of the predefined class but only one cluster had more of data points belong to both the classes. Those records may be the ones whose dissimilarities are very less.
- The above insights will be vital in improving any supervised Learning Algorithms, if we use it for prediction.

Cluster Interpretation of Prefered Solution (f)

Below is the result of the prefered cluster solution

```
## K-means clustering with 2 clusters of sizes 453, 230
## 
## Cluster means:
##         V1       V2       V3       V4       V5       V6       V7       V8
## 1 3.055188 1.298013 1.428256 1.353201 2.094923 1.317881 2.092715 1.260486
## 2 7.173913 6.800000 6.734783 5.739130 5.478261 7.930435 6.108696 6.039130
##         V9
## 1 1.112583
## 2 2.569565
## 
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18 
##   1   2   1   2   1   2   1   1   1   1   1   1   1   1   2   1   1   1 
##  19  20  21  22  23  25  26  27  28  29  30  31  32  33  34  35  36  37 
##   2   1   2   2   1   1   1   1   1   1   1   1   1   2   1   1   1   2 
##  38  39  40  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56 
##   1   2   2   2   2   2   2   1   2   1   1   2   1   1   2   2   2   2 
##  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74 
##   2   1   2   1   2   1   2   1   1   1   1   2   2   1   1   2   1   2 
##  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92 
##   2   1   1   1   1   1   1   1   1   1   2   2   2   2   1   1   1   1 
##  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 
##   1   1   1   1   1   1   2   2   2   1   1   1   2   1   2   2   1   2 
## 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 
##   1   2   2   2   1   1   1   2   1   1   1   1   2   2   2   1   2   1 
## 129 130 131 132 133 134 135 136 137 138 139 141 142 143 144 145 147 148 
##   2   1   1   1   2   1   1   1   1   1   1   1   1   2   1   1   2   1 
## 149 150 151 152 153 154 155 156 157 158 160 161 162 163 164 166 167 168 
##   1   2   1   2   2   1   1   2   1   1   2   2   1   1   1   1   2   2 
## 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 
##   1   1   1   1   1   2   2   2   1   2   1   2   1   1   1   2   2   1 
## 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 
##   2   2   2   1   2   2   1   1   1   1   2   1   1   1   2   2   1   1 
## 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 
##   1   2   2   1   1   1   2   2   1   2   2   2   1   1   2   1   1   2
## 223 224 225 226 227 228 229 230 231 232 233 234 235 237 238 239 240 241
##   1   2   2   1   2   2   1   2   2   2   1   2   1   2   2   2   2   1
## 242 243 244 245 246 247 248 249 251 252 253 254 255 256 257 258 259 260
##   1   1   1   1   1   2   2   1   1   2   2   2   2   2   1   1   1   2
## 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 277 278 279
##   2   2   2   2   2   1   2   2   2   1   2   1   2   1   1   1   1   1
## 280 281 282 283 284 285 286 287 288 289 290 291 292 294 296 297 299 300
##   2   1   1   2   2   2   2   2   1   2   2   1   1   2   2   2   1   2
## 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 317 318 319
##   2   1   2   1   2   2   1   1   2   1   1   1   2   1   1   2   2   1
## 320 321 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338
##   2   2   1   2   1   1   2   1   2   2   2   1   1   2   2   1   2   1
## 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356
##   1   2   2   1   1   1   2   1   1   1   1   2   1   1   2   2   1   1
## 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374
##   1   2   2   2   2   2   1   1   1   1   2   2   1   1   1   1   1   1
## 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392
##   1   1   1   1   1   1   1   2   1   1   1   1   2   1   1   1   1   2
## 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410
##   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1   1
## 411 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429
##   1   2   1   2   1   2   1   1   1   1   2   1   1   1   2   1   2   1
## 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447
##   1   1   1   1   1   2   2   2   1   1   1   2   1   1   1   1   1   1
## 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465
##   1   1   2   1   1   1   2   1   1   2   2   1   1   1   1   1   1   1
## 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483
##   2   2   2   1   1   1   1   1   1   1   1   1   1   1   2   1   1   2
## 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501
##   2   1   1   1   2   2   1   1   2   1   2   1   1   1   1   1   1   1
## 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519
##   1   1   1   1   1   2   1   1   1   1   1   1   1   2   2   1   1   1
## 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537
##   2   1   1   2   2   1   1   1   1   1   1   2   1   1   1   1   1   1
## 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555
##   1   1   1   1   1   1   1   1   1   2   1   1   2   1   1   1   1   1
## 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573
##   1   1   1   1   1   1   1   1   1   1   2   1   1   2   2   2   2   1
## 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591
##   1   2   1   1   1   1   1   1   2   2   1   1   1   2   1   2   1   2
## 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609
##   2   2   1   2   1   1   1   1   1   1   1   1   2   2   2   1   1   2
## 610 611 612 613 614 615 616 617 619 620 621 622 623 624 625 626 627 628
##   1   2   2   2   1   1   1   1   1   1   1   1   1   1   1   1   2   1
## 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646
##   1   1   1   1   1   2   1   1   2   1   1   1   1   1   1   1   1   1
## 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664
##   1   1   2   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1
## 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682
##   1   1   1   1   2   2   2   1   1   1   1   1   1   1   1   1   2   2
## 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699
##   1   1   1   1   1   1   1   1   1   2   1   1   1   1   2   2   2
##
```

```
## Within cluster sum of squares by cluster:
## [1]  4384.565 14938.609
##  (between_SS / total_SS =  60.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Here Ratio of between_SS/total_SS is slightly more than 60%, which indicates that it is fairly good cluster but not very good. between_SS represents devience between clusters and tot_SS represents deviance within the clusters. So, it would be ideal for tot_SS to be very less and between_SS to be high. More the percentage of between_SS/tot_SS, more the internal cohesion and external spearation.

Conclusion

In this exercise, we performed various clustering methods for the selected dataset and analysed the result of each clustering methods.

We found that the density based clustering is not suitable for this dataset, same density based clustering would have been very good in the dataset which have nosie and clusters of different size.

We found that the K-Means clustering gave better results and we concluded based on the comparison with the exixting classes in the data.