dog_analysis.R

sayal

Tue Apr 21 15:23:38 2020

```
#Loading the dataset
df <- read.csv("C:/Users/sayal/Downloads/Final_Data.csv")
View(df)
names(df)
```

```
##  [1] "CanineGroup" "X1"          "X2"          "X3"          "X4"
##  [6] "X5"          "X6"          "X7"          "X8"          "X9"
## [11] "Gender"
```

```
summary(df)
```

```
##        CanineGroup       X1            X2                X3
##  Cuons       :17   Min.   :105   Min.   : 7.200   Min.   : 2.00
##  GoldenJackal:20   1st Qu.:114   1st Qu.: 8.700   1st Qu.:19.00
##  IndianWolves:14   Median :125   Median :10.000   Median :21.00
##  ModernDog   :16   Mean   :129   Mean   : 9.961   Mean   :21.64
##  ThaiDogs    :10   3rd Qu.:137   3rd Qu.:10.900   3rd Qu.:25.00
##                    Max.   :177   Max.   :13.400   Max.   :32.00
##        X4               X5               X6             X7
##  Min.   :15.00   Min.   :17.00   Min.   : 6.0   Min.   :26.00
##  1st Qu.:18.00   1st Qu.:19.00   1st Qu.: 7.1   1st Qu.:30.00
##  Median :22.00   Median :20.00   Median : 7.9   Median :31.00
##  Mean   :21.49   Mean   :20.49   Mean   : 8.0   Mean   :32.52
##  3rd Qu.:24.00   3rd Qu.:22.00   3rd Qu.: 8.7   3rd Qu.:33.00
##  Max.   :28.00   Max.   :27.00   Max.   :10.5   Max.   :43.00
##        X8              X9              Gender
##  Min.   :31.0   Min.   :4.300   Female :32
##  1st Qu.:34.0   1st Qu.:5.300   Male   :35
##  Median :36.0   Median :6.100   Unknown:10
##  Mean   :37.4   Mean   :6.075
##  3rd Qu.:39.0   3rd Qu.:6.800
##  Max.   :50.0   Max.   :8.500
```

```
names(df)[1]= 'CanineGroup'

#Converting CanioneGroup to factor variable
df$CanineGroup <- as.factor(df$CanineGroup)

#Printing the levels of Canine Group
levels(df$CanineGroup)
```

```
## [1] "Cuons"        "GoldenJackal" "IndianWolves" "ModernDog"
## [5] "ThaiDogs"
```

```
#Changing Gender to Factor variable
df$Gender <- as.factor(df$Gender)

#Printing the levels of Gender to Console
levels(df$Gender)
```

```
## [1] "Female"  "Male"    "Unknown"
```

```
#Checking different groups for Canine
levels(df$CanineGroup)
```

```
## [1] "Cuons"        "GoldenJackal" "IndianWolves" "ModernDog"
## [5] "ThaiDogs"
```

```r
#Question1

#Assigning different colors as per CanineGroup
my_cols <- c("#FF0000", "#0000FF","#228B22","#00FF00","#A9A9A9")
#Printing Scatterplot for X1 to X9
pairs(df[,c(2:10)], main='Draftsman plot' ,pch = c(1,16,9,12,14)[as.numeric(df$CanineGroup)], cex = 0.5, col
= my_cols[df$CanineGroup], )
legend(-0.003,1.07,c("Cuons","GoldenJackal","IndianWolves","ModernDog","ThaiDogs"),pch=c(1,16,9,12,14), cex=
0.7, text.font=2)


#Question2

#Calculating Distance Matrix

dist.df <- dist(df[,c(2:10)],method='euclidean')
dist.df
```

```
##            1          2          3          4          5          6
## 2  16.1953697
## 3   8.4486685 16.5990964
## 4   8.2740558  9.2320095 11.3727745
## 5  28.3190748 14.7566934 29.6251582 20.4799902
## 6   5.6435804 13.2385800  6.9649121  5.7105166 25.5479941
## 7   4.9091751 12.1420756  7.3389373  6.3150614 25.5035292  4.5705580
## 8   7.2532751 12.7381317  5.9506302  7.8262379 26.5190498  4.3749286
## 9   4.5144213 17.2130764  7.2291078  9.9829855 30.2504545  5.4817880
## 10  5.0487622 16.0168661  6.0852280  9.6628153 29.5645057  5.2687759
## 11 10.0224747 23.0560187  7.9315824 16.4720976 36.3546421 11.2285351
## 12 12.3567795 26.1568347 11.8789730 19.4650970 39.6598033 14.6301059
## 13  4.9132474 13.9882093  7.6118329  7.6000000 27.3895966  4.6010868
## 14  7.4417740  9.7514102  8.8701747  3.5185224 22.6099536  3.7881394
## 15 10.5441927  8.4669947 11.6086175  4.6540305 20.8281060  6.3804389
## 16  5.1205468 12.8339394 10.4278473  3.9648455 23.7611027  4.8836462
## 17  8.9112289 18.5461586  7.0576200 13.2872119 32.4709101  8.4581322
## 18 18.3185698 31.1412909 15.6815178 24.9200722 44.7600268 19.5754949
## 19 15.8208723 28.8955014 14.4672043 22.5353056 42.5735834 17.3496398
## 20  9.3733665 22.6033183  9.3155784 15.9662143 36.0353993 10.9110036
## 21 11.2191800 24.4225306 10.1098961 17.9134028 37.9465413 12.7334206
## 22 14.7448296 27.7189466 13.7829605 21.2671108 41.3702792 16.1300961
## 23 13.0403221 25.2812183 11.1843641 19.3693056 39.2038263 14.4006944
## 24  9.4741754 21.6520207  9.4899947 15.2413910 35.2535105 10.3193992
## 25 10.3281170 23.8407215  9.6234090 16.7991071 37.0170231 11.6661905
## 26 13.5225737 26.3334388 12.2237474 19.9022612 39.9781190 14.7353317
## 27 15.3195953 28.4193596 13.6890467 22.0374681 42.0844389 16.8404275
## 28 13.8629001 27.7220129 13.9133030 20.9380037 41.2163802 15.9590100
## 29 19.4283298 31.5469491 17.6312223 25.6963032 45.5017582 20.6177108
## 30 17.8308721 30.7351590 16.7008982 24.4830554 44.5453701 19.4069575
## 31 16.3993902 29.0666476 15.4447402 22.7982455 42.7760447 17.6343415
## 32 19.9032661 33.6313842 19.2213423 26.9755445 47.2303928 21.9401459
## 33 18.3885834 31.9882791 17.7690743 25.4899980 45.7065641 20.5370397
## 34 18.5010810 32.4154284 17.5467946 25.7011673 45.9055552 20.5684224
## 35 15.4706173 27.5733567 13.5018517 21.6919340 41.5033734 16.5653252
## 36 14.6266879 27.1685480 13.0873985 20.8868380 40.8530293 15.6904430
## 37  6.3568860 15.7511904  9.4482803  9.0027773 28.6513525  6.5635356
## 38 14.1545046  8.3510478 16.4124952  6.4536811 16.0480528 11.2312065
## 39 16.6439178  8.3624159 19.5923454  9.3091353 13.1609270 14.7665162
## 40 19.6870516  9.4957885 22.7499451 12.1876987 10.8871484 17.6881316
## 41 14.2797059  8.1767964 17.3092461  7.0887234 15.7689568 12.3911259
## 42 14.8101317  5.2962251 16.4024388  7.3443856 14.5989726 12.3527325
## 43  9.7642204  9.5629493 12.7687118  4.3058100 20.3147729  8.1160335
## 44 15.7260930  8.1080207 19.0633156  8.5164547 14.8653961 13.8744369
## 45 14.2551745  8.0857900 17.2867001  7.0915443 15.8221364 12.3174673
## 46  9.3536089 10.7861022 14.2776048  4.5265881 20.6489709  8.9140339
## 47  9.1219515  9.2811637 11.8528478  4.1484937 21.1362248  7.5193085
## 48 23.4296820 10.5517771 25.1827322 15.7305435  7.4793048 20.9652093
## 49 27.3704220 17.6553108 24.7511616 24.0765446 26.0074989 25.8094944
## 50  6.2593929 16.8869772 11.3384302  9.7948966 29.2904421  8.1651699
## 51 16.7349933 10.2562176 19.8479218  9.6104110 13.9495520 14.9026843
## 52  7.8625696 10.9366357 10.3846040  4.8600412 23.0453900  6.3364028
## 53  7.0149840 16.9487463 10.2190998 10.2844543 29.7196904  8.0423877
## 54 47.5963234 34.4450287 48.9106328 40.0610784 20.1697794 44.9966665
```

```
## 54 47.5965254 34.4450287 48.9106528 40.0610784 20.1697794 44.9966665
## 55 45.3097120 31.7889918 45.9642252 37.6426620 17.7578152 42.3584702
## 56 32.1344052 18.5243084 31.3678179 24.6852182  9.1471307 28.7417814
## 57 24.9190690 14.0644943 26.6063902 17.2261429  6.4412732 22.0909484
## 58 59.0581916 45.5361395 59.9957498 51.3385820 31.2931302 56.1538957
## 59 47.6620394 34.8795069 49.0103050 39.9601051 20.2894061 44.8898652
## 60 44.6508679 31.2235488 45.4424911 37.0097285 17.4398968 41.6913660
## 61 45.0948999 32.3020123 46.6871503 37.5453060 18.0427271 42.5109398
## 62 13.4052229  9.9664437 11.5524889  8.7965902 19.2927448 10.5038088
## 63 43.8224828 29.7798590 44.5161768 35.9292360 15.9135163 40.6588244
## 64 45.3348652 31.3263467 45.4287354 37.6755889 18.4092368 42.1072440
## 65 22.0558382  8.9938868 21.1442664 14.8357676 10.4484449 18.6040318
## 66 27.6481464 15.7686398 28.7019163 20.1300770  7.3736016 24.5542257
## 67 38.6082893 24.9953996 39.1573748 30.9977419 11.5883562 35.6181134
## 68 13.5295972 26.3677834 11.6709040 20.1843999 40.0841615 15.1861779
## 69 10.0084964 22.4543982  7.0064256 16.2637634 35.6050558 11.6335721
## 70 15.3521985  6.8716810 16.2138829  8.5445889 13.9706836 12.9514478
## 71 13.4305622 26.9553334 12.8163957 20.3312567 40.4760423 15.5695215
## 72  9.1049437 11.3969294 12.4338248  6.4171645 20.8597699  9.3096724
## 73  6.9433421 13.1461021  6.9649121  8.2103593 25.6082018  7.3348483
## 74 12.0357800  6.8007353 11.4350339  6.9957130 19.6583316  8.6434947
## 75  3.5397740 16.9487463  5.9405387  9.7984693 29.7203634  6.0448325
## 76  2.7110883 16.1598267  6.9404611  8.8232647 28.7982638  5.3047149
## 77  5.2497619 13.5310753  6.0514461  7.6511437 26.5941723  5.3037722
##             7          8          9         10         11         12
## 2
## 3
## 4
## 5
## 6
## 7
## 8   3.9812058
## 9   5.9413803  5.7297469
## 10  4.5923850  4.1109610  2.5787594
## 11 11.6245430 10.5517771  7.5213031  7.6052613
## 12 14.4903416 13.8701118  9.8635693 10.3633971  4.9436828
## 13  3.4928498  3.9761791  3.6578682  3.0740852 10.3677384 12.6273513
## 14  4.2825226  4.8135226  7.8268768  7.1770467 14.1658039 17.2206272
## 15  7.4578817  7.3409809 10.4775951  9.9171569 16.8686099 19.7856008
## 16  5.6356011  7.9429214  7.8523882  8.0579154 14.1566239 16.9224703
## 17  8.1945104  6.2048368  5.4157179  4.4833024  6.2880840  9.1640602
## 18 19.9682248 18.6091375 15.4677083 15.7327684  8.7772433  6.7557383
## 19 17.5442298 16.3085867 12.8786645 13.2245983  7.0121323  4.9406477
## 20 11.0526015 10.1906820  6.3103090  6.8854920  3.6565011  5.1739733
## 21 12.8891427 11.9004202  8.3719771  8.6567892  2.8600699  3.8078866
## 22 16.4854481 15.1340675 11.4695248 12.0374416  6.5099923  4.3150898
## 23 14.1537981 12.8903064  9.8969692  9.8802834  4.8723711  3.2954514
## 24 10.6066017  9.2330927  5.6727418  6.1359596  4.9689033  6.1098281
## 25 12.4551194 11.4411538  7.2145686  8.1498466  3.5411862  3.8652296
## 26 15.1148933 13.7822349 10.2293695 10.6531685  5.2124850  3.2939338
## 27 17.0578428 15.8234004 12.3583980 12.7342059  6.3118935  3.9446166
## 28 16.0810447 15.3469867 11.0932412 11.8190524  6.5398777  3.2140317
## 29 20.7966343 19.2808195 15.9899969 16.4149322 11.0855762  7.8185676
## 30 19.5780489 18.2540406 14.6089014 15.1825558  9.5283787  6.2377881
## 31 18.0055547 16.6015060 13.0069212 13.6106576  8.4196199  6.0704201
## 32 22.1729565 21.1624668 17.0745425 17.8608510 11.8190524  8.0628779
## 33 20.4846284 19.5445645 15.6652482 16.2225152 10.3058236  6.5030762
## 34 20.7987980 19.9384052 15.9062881 16.6090337 10.1597244  6.7334983
## 35 16.6547291 15.0728232 12.0482364 12.2723266  6.8330081  4.8795492
## 36 16.1263759 14.7299016 11.1669154 11.7093979  6.4953830  4.1844952
## 37  6.4660653  6.6407831  5.1662365  4.9839743 10.8314357 12.3612297
## 38 12.0337027 12.8522372 15.8041134 15.2335157 21.8362085 24.9915986
## 39 14.5182644 16.1663230 18.9261724 18.3240279 25.2287534 28.0515597
## 40 17.4201033 18.9665495 21.7572976 21.2289896 28.3786892 31.0385244
## 41 12.2821008 13.8744369 16.2582287 15.8145503 22.7626888 25.3294295
## 42 11.8177832 13.0923642 16.6571306 15.7120973 22.6068574 25.6076161
## 43  7.7987178  9.5545801 11.8785521 11.0304125 17.6666352 20.4936576
## 44 13.5236829 15.0844291 17.6332073 17.0885927 24.2680860 26.7787229
## 45 12.1954910 13.8181041 16.2175831 15.7365816 22.7147529 25.2871509
## 46  8.4693565 10.7907368 11.8873883 11.6215317 18.3885834 20.7378880
## 47  6.8212902  8.2243541 10.6766099  9.8823074 16.7648442 19.3871091
## 48 20.6254697 21.8398718 25.3252838 24.5409861 31.5566158 34.4554785
## 49 23.0139088 22.8335280 27.7751688 25.5861291 30.3731790 32.9072940
```

```
## 50  7.5405570  8.7022985  6.2753486  6.4861391 11.7961858 12.7275292
## 51 15.1419946 16.8846084 19.0929306 18.7245828 25.3931093 28.0750779
## 52  5.8702640  7.3273460  9.2227978  8.3204567 14.9769823 17.5348225
## 53  7.6922038  8.1104870  5.9674115  6.0299254 11.0072703 11.9398492
## 54 45.0294348 46.1978354 49.8881750 49.1794673 55.7065526 59.2332677
## 55 42.5587829 43.4269502 47.3688716 46.6064373 52.9603625 56.6536848
## 56 28.9309523 29.3899643 33.8340066 32.7946642 38.7051676 42.5982394
## 57 22.7292763 23.8631515 27.0935417 26.5625676 32.9613410 36.4281485
## 58 56.4149803 57.3164898 61.1474448 60.4441891 66.9018684 70.5261654
## 59 45.2960263 46.3238599 49.8502758 49.2891469 55.7524887 59.2514979
## 60 41.8998807 42.7309022 46.6962525 45.9161192 52.2732245 56.0190146
## 61 42.6503224 43.7874411 47.3895558 46.7390629 53.2715684 56.7741138
## 62 10.7694011 11.3661779 15.0572242 14.0531135 18.9538914 22.8273958
## 63 40.9026894 41.6261937 45.6167732 44.8537624 51.4340354 55.0023636
## 64 42.2607383 42.8830969 47.1604707 46.2218563 52.4696103 56.2958258
## 65 18.5983870 19.0654137 23.5333805 22.4283303 28.4722672 32.2302653
## 66 25.0291830 25.8536264 29.6070937 28.8664165 35.1162356 38.8863729
## 67 35.7135829 36.5886594 40.6135445 39.7739865 46.1385956 49.8506770
## 68 15.0708328 13.8332932 10.7475579 10.8839331  4.5978256  3.4205263
## 69 11.4083303 10.9644881  8.4077345  8.3270643  4.6626173  6.4078077
## 70 12.6526677 14.0171324 17.5781114 16.7779617 22.9891279 26.4775376
## 71 15.5473470 14.7271857 10.6855042 11.3688170  6.2489999  2.2737634
## 72  8.2571181 11.1341816 12.5499004 12.1387808 17.7569705 20.7824445
## 73  5.2924474  7.2677369  8.7091905  7.6967526 12.6633329 15.4350251
## 74  8.0808415  8.0628779 12.7318498 11.4764977 17.7400676 21.4967439
## 75  5.4744863  6.4187226  3.7134889  3.9268308  7.8243211 10.3469802
## 76  4.7968740  6.2593929  3.4307434  4.0074930  8.9677199 11.4096450
## 77  3.5242020  5.2048055  5.9581876  5.0447993 10.8779594 13.5266404
##           13         14         15         16         17         18
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14  5.1322510
## 15  7.7420927  3.5327043
## 16  6.5084560  5.1322510  7.4471471
## 17  6.8782265 10.4196929 12.7078716 12.1239433
## 18 18.3032784 22.4140581 24.7505555 22.6117226 13.2325357
## 19 15.8139179 20.1400099 22.5019999 20.2059397 10.5957539  4.1725292
## 20  9.3957437 13.6227750 16.1635392 13.6937942  5.0348784  9.8351411
## 21 11.3582569 15.5438091 18.0800996 15.5540991  6.6196677  7.5458598
## 22 14.4183910 18.8417091 21.0876741 18.9939464  9.5467272  5.5190579
## 23 12.3470644 16.7871975 19.1700287 17.3150224  7.5802375  6.5467549
## 24  8.3940455 12.7835832 15.0179892 13.3018796  4.2720019 10.8466585
## 25 10.3542262 14.5358178 16.9496313 14.3627992  7.2594766  8.6313383
## 26 13.0560331 17.4338751 19.6667232 17.6889796  8.4291162  5.8455111
## 27 15.2646651 19.6028059 21.9979545 19.7060904 10.3043680  3.6769553
## 28 14.0968081 18.6799358 21.0366347 18.3684512  9.9914964  6.2265560
## 29 18.5002703 23.0047821 24.9667779 23.6249868 13.7931142  5.3712196
## 30 17.4086186 21.9763509 24.1594702 22.1995495 12.6257673  4.7528939
## 31 15.8745079 20.2716551 22.3459169 20.6291057 10.7754350  5.7280014
## 32 20.0339711 24.6450806 26.9135654 24.3975409 15.6764154  5.5722527
## 33 18.5218790 23.1814581 25.5432966 22.9760745 13.9817739  4.9889879
## 34 18.9280216 23.3856794 25.7784794 23.0744447 14.4720420  4.0570926
## 35 14.6785558 19.0499344 21.2673459 19.6789227  9.3962759  5.0606324
## 36 13.9305420 18.2877008 20.3892128 18.7483333  9.3257707  5.4616847
## 37  4.0012498  7.4289972  9.2752358  7.5133215  8.6429162 18.1802090
## 38 13.3809566  9.1372862  8.1123363  9.4725920 18.5790204 30.2003311
## 39 16.1015527 12.0722823 11.0063618 12.0141583 22.0147678 33.6637788
## 40 18.8005319 14.7458469 13.0782262 15.0897316 24.7574231 36.7057216
## 41 13.3794619  9.6109313  8.5492690  9.7411498 19.6341539 31.0177369
## 42 13.6227750  9.3445171  8.4498521 10.6348484 19.1209309 30.9527059
## 43  9.3520051  6.4093681  7.0682388  5.8736701 14.9093930 26.0074989
## 44 14.6673106 11.0059075  9.6088501 11.2840596 20.7398168 32.5026153
## 45 13.3075167  9.5304774  8.3564346  9.7329338 19.5484015 30.9560979
```

```
## 46  9.4873600  7.2006944  7.8644771  5.2163205 15.7334040 26.6844524
## 47  7.7980767  5.2697249  6.2297673  5.7835975 13.7746143 25.0267857
## 48 22.2272355 17.8630904 16.0315314 18.8978835 27.8564176 39.8278797
## 49 24.9415316 23.3096547 23.4823338 26.4159043 26.4966036 36.5027396
## 50  5.5910643  8.9688349 10.8461975  7.6876524 10.1946064 18.9185095
## 51 16.4790776 12.5227792 11.3956132 11.9498954 22.5873859 33.6881285
## 52  6.5772335  5.0239427  6.4156060  5.2440442 12.3729544 23.0865762
## 53  5.2545219  8.8651001 10.6803558  8.5305334  9.5173526 17.9053065
## 54 47.1856970 42.3623654 40.5759781 43.0098826 51.9821123 64.1468627
## 55 44.7008948 39.7611620 37.9593203 40.7329105 49.2351500 61.3116628
## 56 31.2038459 26.2514761 24.7568172 27.9306283 35.1577872 46.8240323
## 57 24.6434575 19.7924228 18.1928557 20.1022387 29.6723777 41.4126792
## 58 58.4630653 53.5560454 51.5814889 54.3791320 63.1045165 75.2413450
## 59 47.2420364 42.3637817 40.4609688 42.9009324 52.1699147 64.1521629
## 60 44.0740286 39.1568640 37.3724497 40.0978802 48.4585390 60.6551729
## 61 44.7438264 39.9556004 38.1865159 40.4925919 49.5392773 61.7522469
## 62 13.1962116  9.2206290 10.2615788 10.6705201 16.6700330 27.1088546
## 63 42.8475203 37.9311218 35.8442464 39.1937495 47.4113910 59.6836661
## 64 44.4874140 39.5404856 37.7010610 40.9136896 48.5310210 60.6578931
## 65 20.8973683 16.0199875 14.8189068 18.1675535 24.8010080 36.5697963
## 66 27.2246212 22.3248740 20.7807603 23.1702395 31.4103486 43.5062065
## 67 37.9323081 33.0009091 31.2585988 34.1312174 42.3430041 54.4710015
## 68 13.4569685 17.7845439 20.3973037 17.9234483  8.7441409  5.8804762
## 69 10.4033648 13.9746199 16.8181450 14.0024998  8.4569498 10.5612499
## 70 15.0322986 10.4273678 10.3503623 11.3934192 19.9042709 31.5141238
## 71 13.3955216 18.0382926 20.5060967 17.8314329 10.1975487  6.7889616
## 72 10.7154095  8.1092540 10.1350876  6.2369865 15.9176003 26.3057028
## 73  6.9491007  6.9476615  9.8868600  7.3600272 11.1991071 20.7316184
## 74 10.3469802  6.3623895  6.9166466  9.5425364 13.5723985 25.9974999
## 75  4.6743984  8.0255841 11.2111552  7.3273460  7.3959448 16.0978259
## 76  3.9051248  7.0915443 10.1690708  6.2553977  7.7575769 17.2063942
## 77  4.0669399  5.4827001  8.5052925  6.4976919  8.7572827 18.9844673
##            19         20         21         22         23         24
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20  6.8673139
## 21  4.8795492  2.6248809
## 22  2.2912878  5.8386642  4.3289722
## 23  4.3783559  4.9000000  3.3136083  3.7629775
## 24  7.7858847  2.7018512  4.2508823  6.2521996  5.2354560
## 25  6.7238382  3.6510273  3.4117444  5.3907328  4.6324939  4.2130749
## 26  3.3555923  4.9335586  3.3481338  2.0469489  2.5357445  5.2249402
## 27  1.4866069  6.5612499  4.4654227  2.2405357  3.6193922  7.4919957
## 28  3.3704599  5.5533774  4.2485292  2.6551836  4.3370497  6.5253352
## 29  5.8326666 10.8581766  9.2612094  5.6559703  7.2048595 10.9316056
## 30  3.6578682  9.0807489  7.4866548  3.5341194  6.0406953  9.4688965
## 31  3.1112698  7.3130021  6.1163715  2.3853721  5.4267854  7.6980517
## 32  5.9531504 11.5948264  9.9161484  6.3757353  8.8549421 12.3247718
## 33  4.1303753  9.9989999  8.2807005  4.9305172  7.0647010 10.7489534
## 34  4.7947888 10.2171425  8.3988094  5.7148928  7.7678826 11.4057003
## 35  2.8460499  6.5589633  4.9000000  2.2605309  3.0049958  6.8073490
## 36  3.8236109  6.0149813  4.6443514  2.5159491  3.4885527  6.3103090
## 37 15.7483332 10.1926444 11.7158013 14.2527190 12.3361258  8.6861959
## 38 27.8152836 21.5844852 23.3657870 26.5808202 24.7058697 20.5871319
## 39 31.3250698 24.9355168 26.7449061 30.0717475 27.9698051 24.0353906
## 40 34.2768143 27.8190582 29.7497899 32.9370612 30.8915846 26.8063425
## 41 28.7675164 22.4109348 24.2264318 27.4069334 25.3112623 21.4184500
```

```
## 41 28.7675164 22.4109348 24.2264318 27.4069334 25.3112625 21.4184500
## 42 28.7718613 22.4646389 24.2026858 27.5951083 25.2406418 21.5652498
## 43 23.6698120 17.5379588 19.1668985 22.5479489 20.3963232 16.6835248
## 44 30.0321494 23.6873384 25.5710774 28.6722863 26.6356903 22.5829582
## 45 28.7104511 22.3501678 24.1615397 27.3550727 25.2582660 21.3562637
## 46 24.1464283 17.8317133 19.6687570 22.8726911 20.9828501 17.0026469
## 47 22.7176143 16.5737745 18.2444512 21.4336185 19.2135369 15.5206314
## 48 37.6601912 31.2936096 33.0871576 36.3859863 34.0848940 30.2924083
## 49 34.9319338 30.7759646 31.6096504 34.4373344 31.2027242 29.9269110
## 50 16.2696036 10.8857705 12.4036285 14.8340824 13.1411567  9.6218501
## 51 31.4966665 25.1523359 26.9297234 30.2266439 28.1959217 24.3174834
## 52 20.8992823 14.9137520 16.4356320 19.6817174 17.4014367 13.9118654
## 53 15.6028843 10.4484449 11.7618026 14.0946798 12.1400165  9.0072193
## 54 61.9884667 55.4708031 57.3398640 60.9393141 58.8176844 54.8733997
## 55 59.2898811 52.8686107 54.6722965 58.2516953 56.1047235 52.2517942
## 56 45.1202837 38.9907681 40.5759781 44.2050902 41.8455493 38.4320179
## 57 39.2542991 32.7948167 34.6223916 38.1325320 36.1940603 32.1496501
## 58 73.1525119 66.7158902 68.5582964 72.0775971 69.9975714 66.0320377
## 59 62.0371663 55.5475472 57.4233402 60.9102619 58.9072152 54.8703016
## 60 58.5132464 52.1129542 53.9393178 57.4951302 55.4300460 51.4837839
## 61 59.4798285 52.9661212 54.8929868 58.4087322 56.4097509 52.3258063
## 62 25.6480019 19.6982233 21.0411502 24.8803135 22.4140581 19.6096915
## 63 57.5846334 51.1760686 53.0288601 56.4597202 54.3457450 50.3933527
## 64 58.6509164 52.3928430 54.1392649 57.7006066 55.5121608 51.7591538
## 65 34.7882164 28.6276091 30.2266439 33.8651739 31.4294448 28.0916358
## 66 41.2405141 34.8703312 36.7065389 40.2572975 38.3459255 34.2995627
## 67 52.4341492 46.0262968 47.8137010 51.4284940 49.2439844 45.4387500
## 68  3.9458839  5.5794265  3.6249138  3.8157568  2.7129320  6.3882705
## 69 10.0394223  6.9591666  6.4853681  9.5781000  6.9785385  7.9981248
## 70 29.5413270 23.0826775 24.8124969 28.5222720 26.2160256 22.6876618
## 71  5.5353410  6.5084560  5.2239832  4.4821870  4.0607881  6.9670654
## 72 24.2738542 17.8297504 19.4856357 23.3420222 21.1728600 17.9058650
## 73 19.1575051 13.2698907 14.5237736 18.2543146 15.5054829 13.2351804
## 74 23.7827669 17.5379588 19.2210822 22.7712538 20.5419084 16.9428451
## 75 14.0744449  7.9479557  9.4371606 12.9799846 10.8369737  8.0280757
## 76 15.0163245  8.5609579 10.3227903 13.8036227 11.8987394  8.5340494
## 77 17.1435119 10.9517122 12.4735721 16.0390149 13.4985184 10.6602064
##            25         26         27         28         29         30
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26  3.9962482
## 27  5.8497863  2.6776856
## 28  5.1156622  2.9359837  3.1448370
## 29  9.7185390  6.3874878  5.5434646  6.6166457
## 30  8.2879430  4.9173163  3.6537652  4.5144213  2.8740216
## 31  7.2117959  3.8288379  3.5227830  3.6769553  4.6968074  2.8390139
## 32 10.4331203  7.6459139  5.9539903  6.3560994  4.5276926  3.4785054
## 33  9.3107465  6.1676576  4.3600459  4.8764741  4.6518813  2.7166155
## 34  9.1684241  6.6340033  4.6238512  5.2220686  5.3376025  3.9255573
## 35  6.2136946  2.6907248  2.4596748  4.1976184  4.9436828  3.6469165
## 36  4.8713448  1.9235384  3.0248967  3.3256578  5.1400389  4.1713307
```

```
## 37 10.2381639 12.8160056 15.1828851 13.9201293 18.1231896 17.1356354
## 38 22.2925997 25.2065468 27.3923347 26.4652602 31.0214442 29.8665365
## 39 25.6368875 28.6517015 30.8248277 29.6954542 34.3132627 33.2225827
## 40 28.6008741 31.5388649 33.7992603 32.5573340 37.0402484 36.0355380
## 41 22.8735655 25.9509152 28.1918428 27.0061104 31.4364438 30.4566906
## 42 23.2415576 26.1040227 28.2324282 27.3704220 31.6742798 30.7148173
## 43 18.2123584 21.0793738 23.1971981 22.1716035 26.9061331 25.7425717
## 44 24.4032785 27.2758135 29.5577401 28.3014134 32.7580524 31.7472833
## 45 22.8381260 25.8860967 28.1417839 26.9349216 31.3606441 30.4024670
## 46 18.4948642 21.5264953 23.6791047 22.2793626 27.2615113 25.9855729
## 47 17.1154901 19.9957495 22.1616786 21.1709707 25.5614945 24.5008163
## 48 32.0024999 34.9022922 37.1179202 36.1415274 40.2880876 39.4564317
## 49 32.1597575 33.1984939 34.5202839 34.8998567 36.9975675 36.6679697
## 50 11.0276924 13.5462172 15.7920866 14.1626269 18.8143562 17.6283862
## 51 25.5634505 28.7433471 30.9401034 29.7314648 34.3997093 33.3517616
## 52 15.2803796 18.1411135 20.3231395 19.3313217 23.7432096 22.7446697
## 53 10.0945530 12.6574089 14.9679658 13.6209398 17.6954796 16.7806436
## 54 56.5863941 59.5532535 61.5754821 60.7088956 65.2532758 64.2156523
## 55 53.9204970 56.8585086 58.8509983 58.1688061 62.4919995 61.5196717
## 56 39.9485920 42.7399111 44.6122181 44.2854378 48.2371226 47.3964134
## 57 33.6319788 36.7257403 38.8068293 37.8879928 42.5593703 41.4446619
## 58 67.7551474 70.6906642 72.7424223 71.9728421 76.3186085 75.3607988
## 59 56.4535207 59.5161323 61.5956167 60.7220718 65.2184790 64.1972741
## 60 53.2921195 56.1521148 58.1371654 57.4459746 61.8480396 60.8031249
## 61 54.1206061 57.0614581 59.1041454 58.1942437 62.8274621 61.7123164
## 62 20.3442867 23.3649310 24.9937992 24.7834622 29.0213714 28.0024999
## 63 52.2277704 55.0647800 57.1624002 56.4279186 60.5598052 59.6899489
## 64 53.6001866 56.3186470 58.2698893 57.7446967 61.8862667 60.9679424
## 65 29.6386572 32.3891957 34.2699577 33.9025073 37.8726814 37.0275573
## 66 36.1287974 38.9458599 40.9197996 40.1808412 44.8024553 43.6275143
## 67 47.1634392 50.0353875 52.0089415 51.3421854 55.6595005 54.6891214
## 68  5.0774009  3.4510868  3.1400637  4.6314145  7.6491830  5.9203040
## 69  5.7393379  8.0653580  8.8938181  9.0967027 12.7188836 11.7630778
## 70 24.0029165 27.0959407 28.9692941 28.2688875 32.8120405 31.6943213
## 71  4.5044423  3.8961519  4.3829214  3.9799497  6.9526973  5.6373753
## 72 18.6678869 21.9895430 23.6767819 22.6145971 27.8068337 26.4196896
## 73 13.6565003 16.7107750 18.3510218 17.7237129 22.0583318 21.0501781
## 74 18.9886808 21.5086029 23.3749866 22.9133149 27.0196225 25.9507225
## 75  8.2668011 11.5874933 13.3364163 12.3567795 17.2785995 15.9263932
## 76  9.1629690 12.4807852 14.3506097 13.0923642 18.1198786 16.7693172
## 77 11.4677810 14.4962064 16.3881054 15.5338340 19.8881874 18.9034388
##            31         32         33         34         35         36
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32  5.6709788
```

```
## 33  4.6108568  2.6720778
## 34  5.3169540  2.7110883  2.8930952
## 35  3.1717503  6.8774995  5.3665631  6.2521996
## 36  3.2031235  6.8920244  5.9008474  6.0141500  2.6267851
## 37 15.8811209 19.6012755 18.1242931 18.7632620 14.6727639 13.8329317
## 38 28.1753438 32.4464174 30.8818717 31.2662758 26.9801779 26.3182446
## 39 31.6711225 35.7029411 34.1443992 34.5139102 30.4263044 29.6981481
## 40 34.4043602 38.5644914 37.0359285 37.4492991 33.2803245 32.4930762
## 41 28.9794755 32.8975683 31.4466214 31.7826997 27.7504955 26.8955387
## 42 29.1715615 33.3487631 31.7272753 32.0875365 27.7196681 27.1101457
## 43 24.2416996 28.2053186 26.5424189 26.9586721 22.9194241 22.2602785
## 44 30.1783035 34.2714167 32.7275114 33.2213787 29.0305012 28.2639346
## 45 28.9013840 32.8513318 31.4007962 31.7234929 27.6985559 26.8199553
## 46 24.5073458 28.2780834 26.7235851 27.1503223 23.4467482 22.6373585
## 47 23.1302832 27.0386760 25.4548620 25.9472542 21.6930865 21.0781878
## 48 37.8794139 42.0923984 40.5572435 40.9094121 36.5120528 35.8085185
## 49 35.7837952 39.6656022 37.6332300 38.6148935 33.3613549 34.0884145
## 50 16.4760432 19.8247320 18.3101065 19.0538710 15.4912879 14.6424042
## 51 31.8238904 35.6768833 34.2271822 34.4515602 30.6682246 29.7344918
## 52 21.3883146 25.2218160 23.6452955 24.0609642 19.9514411 19.2249837
## 53 15.7673714 19.0759010 17.6694652 18.3330303 14.5151645 13.6157996
## 54 62.3176540 66.8470643 65.2679860 65.3561015 61.1319066 60.4811541
## 55 59.6386620 64.2168981 62.6706470 62.7211288 58.3562336 57.7601073
## 56 45.6385802 50.1605423 48.5989712 48.5912544 44.0946709 43.6263682
## 57 39.6093423 43.9611192 42.4714021 42.5333986 38.4902585 37.7059677
## 58 73.4297624 78.0536354 76.5143777 76.5956918 72.2321950 71.5886164
## 59 62.3105930 66.7821084 65.2908110 65.3502869 61.1719707 60.4312006
## 60 58.8635711 63.5329836 61.9385179 62.0515109 57.6373143 57.1007881
## 61 59.7763331 64.3388685 62.7363531 62.8851334 58.6802352 58.0373156
## 62 26.5243285 30.4719543 28.9682585 28.7605633 24.8201531 24.3010288
## 63 57.7677246 62.4643098 60.9348012 61.0717611 56.5410470 55.9132364
## 64 59.0231311 63.7837754 62.1616441 62.2618663 57.6970537 57.2242082
## 65 35.2944755 39.7922103 38.2123017 38.2440845 33.7238788 33.2499624
## 66 41.6290764 46.3065870 44.6925050 44.8094856 40.5183909 39.9564763
## 67 52.7982954 57.4113229 55.8378008 55.8960643 51.5016505 50.9346640
## 68  5.8386642  8.3288655  6.5871086  7.1763500  3.5482390  4.5967380
## 69 11.3767306 13.5952197 12.3697211 11.8726577  9.3749667  8.7412814
## 70 30.1026577 34.2229455 32.6776682 32.7087144 28.6494328 28.0579757
## 71  6.1514226  7.3375745  6.2817195  6.5901442  4.8682646  4.1737274
## 72 25.0323790 28.5408479 27.0469961 26.9594139 23.7457365 23.0156469
## 73 20.0022499 23.3154455 21.8204033 21.7912827 18.2156526 17.6099404
## 74 24.2004132 28.8156208 27.2025734 27.4468577 22.7011013 22.4330114
## 75 14.7665162 18.1099420 16.6754310 16.6991018 13.3712378 12.5908697
## 76 15.4288690 18.9676040 17.6093725 17.6005682 14.3083891 13.3809566
## 77 17.6448293 21.3049290 19.8509446 19.8849189 16.1015527 15.3225324
##            37         38         39         40         41         42
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
```

```
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38 13.7338997
## 39 16.5157501  5.2163205
## 40 19.2756323  7.9931220  3.8288379
## 41 13.6418474  5.0675438  3.5566838  6.1098281
## 42 14.4661674  4.7296934  4.1496988  6.8614867  4.0261644
## 43  9.4228446  5.5758407  7.9586431 11.2942463  6.2201286  6.2225397
## 44 14.8734663  5.3065997  2.9614186  4.4821870  2.7276363  4.6141088
## 45 13.5882302  5.1942276  3.6510273  6.1049161  0.7348469  4.0435133
## 46  9.4836702  6.6166457  7.9189646 10.8779594  5.7740800  7.4518454
## 47  7.9485848  6.8774995  8.9548869 12.0735248  6.4915329  6.8183576
## 48 22.8565089 11.2249722  7.3763134  5.0882217  9.7200823  9.0978019
## 49 26.1704031 23.5136131 23.1408729 24.2169362 23.4770100 20.2825048
## 50  2.7110883 14.3488676 16.7409677 19.4679223 13.9760509 15.1947359
## 51 16.6291912  5.9707621  2.9563491  4.8600412  3.7536649  5.7043843
## 52  6.2569961  8.4575410 10.8415866 13.9642400  8.2758685  8.6255435
## 53  1.8330303 14.9351933 17.4198163 20.1650688 14.4492214 15.5058054
## 54 48.4080572 35.2485461 32.3306356 29.8799264 35.3035409 34.2544888
## 55 46.0091295 32.8344331 30.2848147 28.0633213 33.1647403 31.8356718
## 56 32.7299557 20.4990244 18.8822668 17.9315365 21.3300258 18.9625948
## 57 25.6267438 12.3834567 10.3951912  9.2097774 13.1434394 12.3069086
## 58 59.5971476 46.2985961 43.6189179 41.0853989 46.5175236 45.4964834
## 59 48.2599213 34.9279258 32.2218870 29.7356688 35.0742070 34.3215676
## 60 45.3983480 32.1018691 29.7539913 27.5648327 32.7464502 31.3320922
## 61 45.9092583 32.5993865 29.8745711 27.4060212 32.9095731 31.9206830
## 62 15.1185317 10.1671038 12.0349491 14.9833241 11.3353430  9.0122139
## 63 44.0679022 31.0338525 28.5567855 26.0263328 31.3007987 30.0714150
## 64 45.9144857 32.8910322 30.7868478 28.7222910 33.6609566 31.8698917
## 65 22.6431005 11.9235062 11.4324101 11.9230868 12.9340636  9.8812955
## 66 28.6326736 15.5064503 14.3784561 13.2604676 17.2223692 15.4544492
## 67 39.3661022 26.4028408 23.9820766 21.9667931 26.8408271 25.2455937
## 68 13.4632834 25.4758709 28.8032984 31.8793036 26.2423322 26.1528201
## 69 10.9590146 21.8096309 24.5782424 27.7685073 21.9412853 21.8341476
## 70 16.7970235  7.4953319  7.1826179  9.5671312  7.7188082  5.2924474
## 71 12.9201393 25.8342796 28.8468369 31.7984276 26.0017307 26.4404992
## 72 12.4575278  9.9824847 10.5138005 13.7171426  9.2935461  9.1148231
## 73  8.7407094 13.3078924 14.8946299 18.0883941 12.5865007 12.1008264
## 74 12.8011718  9.1504098 11.6284135 14.1421356 10.7242715  8.4593144
## 75  6.0844063 15.6658865 18.2863337 21.4105114 15.6875747 15.9301601
## 76  5.9177699 14.9331845 17.4129262 20.4031860 14.7878328 15.2305614
## 77  6.4101482 13.4606835 15.5967945 18.5822496 12.9688087 12.8506809
##            43         44         45         46         47         48
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
```

```
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44  7.3000000
## 45  6.1733297  2.6795522
## 46  3.4132096  6.7985293  5.8017239
## 47  2.9748950  7.8676553  6.5375837  3.9799497
## 48 14.7068011  8.7040221  9.7149370 15.0326312 15.3720526
## 49 22.6993392 23.2467202 23.4787138 24.7493434 22.0719279 22.9584407
## 50  9.8061205 15.0495847 13.9380773  9.2048900  8.6353923 23.3642890
## 51  8.4202138  4.2011903  3.8587563  8.0802228  9.6046864  8.2855296
## 52  3.6331804  9.7226540  8.2115772  5.0009999  2.6172505 17.2026161
## 53 10.4742542 15.7524601 14.4166570 10.3121288  8.9386800 23.7330992
## 54 39.5602073 34.1868396 35.3171347 39.8793179 40.8377277 26.4170400
## 55 37.2761318 32.2139721 33.1933728 37.8563073 38.4512679 24.3125482
## 56 24.4020491 20.9716475 21.3394002 25.7738239 25.4096438 14.0573824
## 57 16.8884576 12.5215814 13.1670042 17.3392618 18.3382115  7.8962016
## 58 50.8473205 45.3876635 46.5405200 51.2556338 52.0793625 37.4534378
## 59 39.5207540 34.0429141 35.1252046 39.7675747 40.7816135 26.4037876
## 60 36.6275852 31.6551734 32.7684299 37.2264691 37.8783579 24.0651200
## 61 37.0681804 31.6910082 32.9378202 37.3151444 38.3992187 24.3376252
## 62  9.5947903 13.0648383 11.4021928 11.6614750 10.0254676 16.1049682
## 63 35.6012640 30.1844331 31.3007987 36.1543912 36.6810578 22.2508427
## 64 37.2876655 32.6291281 33.6562030 38.1968585 38.5014285 24.8921674
## 65 14.9224663 13.1015266 12.9255561 16.5966864 15.7432525 10.0682670
## 66 20.0029998 16.2975458 17.2124955 20.8300264 21.5065106 11.5779964
## 67 30.6864791 25.9243129 26.8374738 31.4033438 31.8705193 18.2458214
## 68 21.2153246 27.6423588 26.2270852 21.7816436 20.0469449 35.0944440
## 69 17.3381083 23.7667835 21.9184853 18.0371838 16.2018517 30.6277652
## 70  9.0183147  8.8306285  7.8089692 10.0816665  9.6643675 10.7879562
## 71 21.4350181 27.5123609 25.9832638 21.5696546 20.0950243 35.1796816
## 72  7.2691127 11.0458137  9.3311307  7.1840100  7.6374079 16.6267856
## 73  8.9682774 14.5279042 12.5857062  9.8519034  7.8064076 20.6131026
## 74  8.5252566 11.5312619 10.6578609 10.2610916  8.1884064 16.0415087
## 75 11.2414412 17.3902271 15.6875747 11.4210332 10.0498756 24.7228639
## 76 10.6531685 16.4304595 14.7438123 10.5157025  9.4710084 23.8537209
## 77  9.0917545 14.7292227 12.9015503  9.6855563  7.7103826 21.5079055
##            49         50         51         52         53         54
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
```

```
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50 27.3177598
## 51 25.2847780 16.7755775
## 52 22.9543024  7.0992957 11.0770032
## 53 26.8642886  2.4556058 17.4025860  7.2159545
## 54 41.2927354 48.8508956 32.7134529 42.6057508 49.4722144
## 55 38.4792152 46.6706546 30.7514227 40.2078351 47.1257891  4.8352870
## 56 25.7930223 33.8192253 19.6824795 26.9484693 33.9206427 19.2803008
## 57 27.5227179 26.1537760 10.5233075 19.9323857 26.7434104 23.3051496
## 58 51.2102529 60.1399202 43.9250498 53.8394837 60.6743768 11.8448301
## 59 42.8024532 48.7217611 32.3371304 42.5404513 49.3001014  4.5409250
## 60 38.0110510 46.0306420 30.3697218 39.6577861 46.5777844  5.6329388
## 61 39.9861226 46.3064790 30.2927384 40.2175335 47.0253123  4.3520110
## 62 21.3536882 16.4711870 12.4201449 10.4048066 16.2052461 37.8427007
## 63 36.9901338 44.7791246 29.0473751 38.4324082 45.1940262  7.1147734
## 64 36.8669228 46.7308249 31.4834877 40.1762368 47.1268501  8.2945765
## 65 20.2509259 23.8570744 12.4539150 17.0751281 23.8501572 28.3197811
## 66 27.3338618 29.3127958 15.1914450 23.0989177 29.9444486 21.2181526
## 67 32.5038459 40.0866561 24.6247843 33.5758842 40.5227097 10.3812331
## 68 32.0600998 14.1523850 29.0676796 18.3480244 13.2778010 59.6335476
## 69 29.2648253 11.9703801 24.5713248 14.3934013 10.7359210 55.0143618
## 70 21.2379378 17.6502125  8.2249620 11.2165057 17.9103322 33.1240094
## 71 33.6083323 13.3319166 28.8506499 18.3210262 12.3276113 60.2080559
## 72 24.1524326 12.6475294 10.7786827  8.1608823 13.2404683 39.5930549
## 73 22.6161447  9.7667804 15.1818971  6.9043465  9.2217135 44.9851086
## 74 19.6158100 14.2337627 13.0728727  9.0088845 14.2179464 38.9102814
## 75 26.7486448  6.9289249 18.3915742  8.5726309  6.4668385 49.2609379
## 76 26.9720225  6.6550733 17.5228422  8.1197291  6.5007692 48.3286664
## 77 23.8951878  7.7781746 15.8335088  6.3890531  7.1140706 46.2174210
##            55         56         57         58         59         60
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
```

```
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56 15.4754645
## 57 21.0030950 11.2578861
## 58 14.1513250 29.4042514 34.4149677
## 59  6.0679486 19.8517002 22.9458057 11.9733036
## 60  2.8653098 15.3218798 20.4289011 14.9869944  7.0604532
## 61  5.9481089 18.0036107 20.8189817 14.6928554  5.2191953  5.2201533
## 62 34.8518292 20.3798921 16.3370744 48.9102239 37.9959208 34.4607023
## 63  4.4384682 14.7224319 19.5777935 15.9705980  7.4793048  4.8856934
## 64  4.7853944 14.6112970 21.6242919 15.7473807 10.0189820  4.3139309
## 65 25.1282709 10.6122571  9.6093704 39.1557148 28.7236836 24.7422715
## 66 18.5986559  9.0675245  5.7113921 32.1498056 21.4077089 17.4911406
## 67  7.0604532  9.2249661 14.8579945 21.0387737 11.5494589  6.7660919
## 68 56.8818073 42.5771065 36.9547020 70.8275370 59.7024288 56.1790886
## 69 52.2660502 37.8780939 32.4975384 66.2620555 55.0921047 51.8083970
## 70 30.4831101 17.0601876 11.4843372 44.4641429 33.3517616 30.0281535
## 71 57.5830704 43.5293005 37.3654921 71.4510322 60.1505611 56.9964911
## 72 37.3458164 24.4691643 17.6742751 51.1477272 39.8433181 36.8692284
## 73 42.4108477 28.4480228 22.9741159 56.3636408 45.2272042 42.0381969
## 74 36.0426137 22.0766845 17.4499284 50.0017000 39.2933837 35.3154357
## 75 46.7394908 32.9977272 26.5798796 60.6239227 49.3005071 46.1871194
## 76 45.8791892 32.3668040 25.6739167 59.7139850 48.3834683 45.3114776
## 77 43.6601649 29.8016778 23.8587510 57.5624009 46.3735916 43.2011574
##            61         62         63         64         65         66
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
```

```
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56
## 57
## 58
## 59
## 60
## 61
## 62 35.7884059
## 63  7.0922493 33.7699571
## 64  8.8848185 34.5240496  5.6973678
## 65 26.5120727 10.5933942 23.7118114 24.4542430
## 66 18.6346452 18.3766156 17.2953751 18.2682785 10.0518655
## 67  9.4482803 28.1606108  6.5977269  7.2890329 18.2991803 12.1235308
## 68 57.1809409 22.9039298 55.2736827 56.3368441 32.2733636 39.0785107
## 69 52.7861724 17.7620382 50.8487955 51.9609469 27.6472422 35.0366950
## 70 30.9182794  6.2729578 29.2523503 30.5417747  7.7723870 14.0303243
## 71 57.7772447 23.7970586 55.8865816 57.2954623 33.1707703 39.9617317
## 72 37.3444775  7.9031639 36.2674785 37.6617843 15.1340675 20.6533290
## 73 42.8679367  9.2417531 41.0626351 42.3544567 18.3420282 25.7827462
## 74 36.6972751  7.2567210 34.4071214 35.4153921 11.9431989 18.7829710
## 75 46.8749400 13.4636548 45.2404686 46.6596185 22.7615905 29.3088724
## 76 45.9360425 13.3030072 44.3226804 45.8543346 22.1668672 28.4332552
## 77 43.9798818 10.8157293 42.0743390 43.5271180 19.5064092 26.6063902
##            67         68         69         70         71         72
## 2
## 3
## 4
## 5
## 6
```

```
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56
## 57
## 58
## 59
## 60
## 61
## 62
## 63
## 64
## 65
## 66
## 67
## 68 50.0532716
## 69 45.4912079  7.0042844
## 70 23.7968485 26.8387779 22.0249858
## 71 50.8163360  3.9306488  7.1239034 27.3921522
## 72 30.7470324 21.5826319 16.5677397  7.7466122 21.7628123
## 73 35.6815078 16.0965835 10.5801701 12.1218810 16.2434602  7.1295161
## 74 29.1352021 21.3046943 17.6672012  7.9303216 22.3950887  9.0509668
## 75 40.0071244 11.2889326  6.9641941 16.3951212 11.2022319 10.6324974
## 76 39.1421767 12.4987999  8.4297094 15.7028660 12.2478570  9.8818015
## 77 36.8814316 14.3669760  9.4493386 13.4376337 14.3436397  8.5732141
##            73         74         75         76
## 2
```

```
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23
## 24
## 25
## 26
## 27
## 28
## 29
## 30
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
## 51
## 52
## 53
## 54
## 55
## 56
## 57
## 58
## 59
## 60
## 61
## 62
## 63
## 64
## 65
## 66
## 67
## 68
## 69
## 70
## 71
## 72
## 73
## 74  9.6948440
```

```
## 75  5.9447456 12.5223800
## 76  6.0909769 11.8545350  1.9544820
## 77  3.1890437  9.7867257  4.0657103  3.7669616
```

```r
#Question3:

#Applying PCA function on the dataset
pca.df <- prcomp(df[,c(2:10)], scale=TRUE)

#Printing the results of pca to console
pca.df
```

```
## Standard deviations (1, .., p=9):
## [1] 2.6555963 0.8391652 0.7365758 0.4390554 0.4241988 0.3627806 0.3031519
## [8] 0.2652189 0.1857418
##
## Rotation (n x k) = (9 x 9):
##          PC1         PC2         PC3         PC4         PC5         PC6
## X1 0.3636408 -0.11451510  0.08210471 -0.30326354  0.24950692 -0.07899550
## X2 0.3424554  0.31490128 -0.19979188  0.33605928  0.01517931  0.49451257
## X3 0.2665621  0.32018675  0.87894338  0.04161625 -0.18169514 -0.04568559
## X4 0.3265349  0.44638084 -0.16540131  0.26534253  0.54545187 -0.21526217
## X5 0.3539586 -0.14160855 -0.03861441 -0.26352534 -0.33012092  0.43239890
## X6 0.3459444  0.06792334 -0.26250857  0.05378069 -0.51974026 -0.68294862
## X7 0.2859405 -0.68736531  0.13651981  0.64014932  0.05443187 -0.01170970
## X8 0.3470802 -0.28877388  0.03666665 -0.47256682  0.40753260 -0.10978603
## X9 0.3544268  0.07362113 -0.25111557 -0.13231892 -0.24254817  0.18765482
##            PC7         PC8          PC9
## X1  0.05543869  0.16005914  0.811637429
## X2  0.13657790  0.60411640 -0.048224206
## X3  0.08257828 -0.03476461 -0.094992855
## X4 -0.30849700 -0.39244126 -0.057608736
## X5 -0.67024916 -0.19081879 -0.046144083
## X6 -0.08734948  0.23986950 -0.046794522
## X7  0.03463451 -0.11415807  0.002559605
## X8  0.12889717  0.23397418 -0.567438948
## X9  0.63372357 -0.54081471 -0.016253801
```

```r
#Printing the summary of the pca to console
summary(pca.df)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6
## Standard deviation     2.6556 0.83917 0.73658 0.43906 0.42420 0.36278
## Proportion of Variance 0.7836 0.07824 0.06028 0.02142 0.01999 0.01462
## Cumulative Proportion  0.7836 0.86182 0.92210 0.94352 0.96352 0.97814
##                            PC7     PC8     PC9
## Standard deviation     0.30315 0.26522 0.18574
## Proportion of Variance 0.01021 0.00782 0.00383
## Cumulative Proportion  0.98835 0.99617 1.00000
```

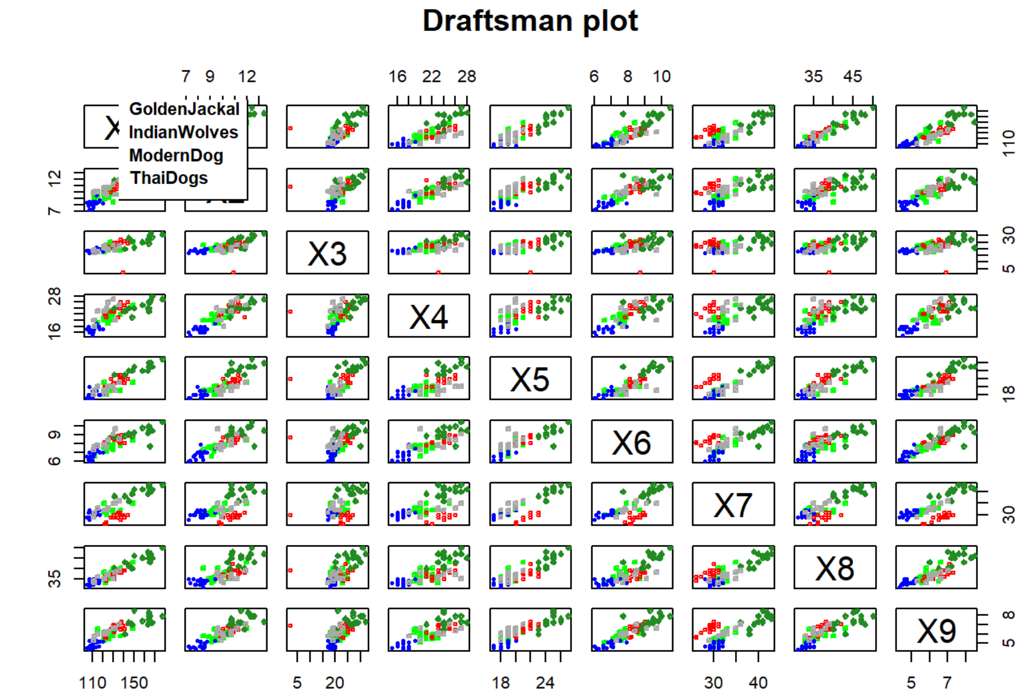
```r
#If we look at PCA summary we get 92% variance in the first 3 columns. Thus, we can use these 3 variables in
stead of X1 to X9

#Plotting PCA

library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.5.3
```

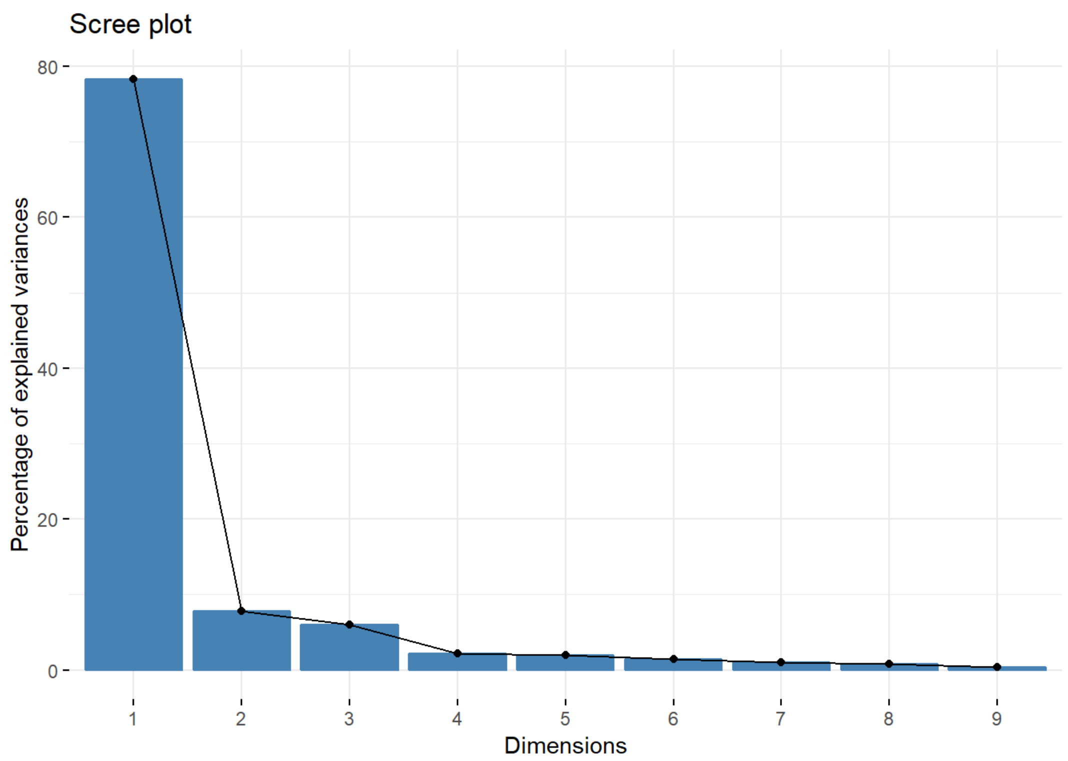
```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.5.2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```




```
#Printing the Scree plot of PCA
fviz_eig(pca.df)
```




```
#Question 4.

#Changing levels of Gender to 0,1,2
levels(df$Gender) <- c(0,1,2)
#Changing Gender from ctegorical to numerical
df$Gender <- as.numeric(df$Gender)
#Creating matrix with scaled values
matstd.df <- scale(df[,2:11])
#Applying Kmeans for predicting 5 groups with 5 random points as starting points
(kmeans5.df <- kmeans(matstd.df,5,nstart = 5))
```

```
## K-means clustering with 5 clusters of sizes 17, 20, 13, 10, 17
##
## Cluster means:
##           X1          X2         X3          X4         X5         X6
## 1  0.5257981  0.82437100  0.6027277  0.82906440  0.6286153  0.6550842
## 2 -1.0326917 -1.28012456 -0.7442101 -1.31541041 -0.9210487 -1.2015623
## 3 -0.1611360 -0.05995349 -0.2422624  0.03607956 -0.2290787  0.0375723
## 4  1.9898442  1.31071721  1.3438925  1.15643855  1.8900795  1.4555510
## 5 -0.3581416 -0.04350544 -0.3324519  0.01062903 -0.4816623 -0.1264198
##            X7         X8          X9      Gender
## 1  0.01647758  0.4294301  0.68760357  0.15936106
## 2 -0.53191468 -0.9427605 -1.25559574 -0.38553117
## 3 -0.62224704 -0.2660391 -0.02106367 -1.04197614
## 4  2.08034976  2.0199694  1.59329540 -0.02083952
## 5 -0.13859478 -0.3050758 -0.13155719  1.10326886
##
## Clustering vector:
##  [1] 5 5 5 5 1 5 5 5 3 3 3 2 3 3 3 3 2 2 2 2 2 2 2 2 5 2 2 2 2 2 2 2 2 2 2
## [36] 2 5 1 1 1 1 1 1 1 1 3 3 1 3 3 1 3 3 4 4 4 1 4 4 4 4 1 4 4 1 1 4 5 5 1
## [71] 5 1 5 5 5 5 5
##
## Within cluster sum of squares by cluster:
## [1] 49.84010 26.74809 47.40541 28.08717 36.88456
##  (between_SS / total_SS =  75.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
kmeans5.df$cluster
```

```
##  [1] 5 5 5 5 1 5 5 5 3 3 3 2 3 3 3 3 2 2 2 2 2 2 2 2 5 2 2 2 2 2 2 2 2 2 2
## [36] 2 5 1 1 1 1 1 1 1 1 3 3 1 3 3 1 3 3 4 4 4 1 4 4 4 4 1 4 4 1 1 4 5 5 1
## [71] 5 1 5 5 5 5 5
```

```
#Applying Kmeans for predicting 5 groups with 10 random points as starting points
(kmeans10.df <- kmeans(matstd.df,5,nstart = 10))
```

```
## K-means clustering with 5 clusters of sizes 19, 10, 20, 18, 10
##
## Cluster means:
##           X1         X2         X3         X4         X5         X6
## 1  0.4465495  0.7255130  0.6108086  0.7108358  0.6049923  0.6375427
## 2  1.9898442  1.3107172  1.3438925  1.1564386  1.8900795  1.4555510
## 3 -1.0326917 -1.2801246 -0.7442101 -1.3154104 -0.9210487 -1.2015623
## 4 -0.2334116 -0.2216926 -0.3487361 -0.1625387 -0.3989820 -0.2496471
## 5 -0.3527640  0.2701040 -0.3882835  0.4163640 -0.4793000  0.1856072
##            X7         X8          X9      Gender
## 1 -0.20017854  0.3268102  0.62293299 -0.27420425
## 2  2.08034976  2.0199694  1.59329540 -0.02083952
## 3 -0.53191468 -0.9427605 -1.25559574 -0.38553117
## 4 -0.39078317 -0.2301413 -0.19917961 -0.31259284
## 5  0.06722854 -0.3411333  0.09284671  1.87555706
##
## Clustering vector:
##  [1] 4 4 4 4 1 4 4 4 4 4 4 3 4 4 4 1 3 3 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3
## [36] 3 4 1 1 1 1 1 1 1 1 1 1 1 4 4 1 1 4 2 2 2 1 2 2 2 2 1 2 2 1 1 2 5 5 5
## [71] 5 5 5 5 5 5 5
##
## Within cluster sum of squares by cluster:
## [1] 49.36404 28.08717 26.74809 56.89736 24.97754
##  (between_SS / total_SS =  75.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
kmeans10.df$cluster
```

```
##  [1] 4 4 4 4 1 4 4 4 4 4 4 3 4 4 4 1 3 3 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3
## [36] 3 4 1 1 1 1 1 1 1 1 1 1 1 4 4 1 1 4 2 2 2 1 2 2 2 2 1 2 2 1 1 2 5 5 5
## [71] 5 5 5 5 5 5 5
```

```
#Applying Kmeans for predicting 5 groups with 15 random points as starting points
(kmeans15.df <- kmeans(matstd.df,5,nstart = 15))
```

```
## K-means clustering with 5 clusters of sizes 18, 12, 20, 10, 17
##
## Cluster means:
##           X1         X2         X3         X4         X5         X6
## 1 -0.2334116 -0.2216926 -0.3487361 -0.1625387 -0.3989820 -0.2496471
## 2  1.8250998  1.2097445  1.3320283  1.0133575  1.8097616  1.3839132
## 3 -1.0326917 -1.2801246 -0.7442101 -1.3154104 -0.9210487 -1.2015623
## 4 -0.3527640  0.2701040 -0.3882835  0.4163640 -0.4793000  0.1856072
## 5  0.3812756  0.7279403  0.5329382  0.7594103  0.5105006  0.5918743
##            X7         X8          X9      Gender
## 1 -0.39078317 -0.2301413 -0.19917961 -0.31259284
## 2  1.93256109  1.8005079  1.49522686 -0.06946508
## 3 -0.53191468 -0.9427605 -1.25559574 -0.38553117
## 4  0.06722854 -0.3411333  0.09284671  1.87555706
## 5 -0.36415458  0.2825290  0.57799756 -0.26968794
##
## Clustering vector:
##  [1] 1 1 1 1 5 1 1 1 1 1 1 3 1 1 1 5 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3
## [36] 3 1 5 5 5 5 5 5 5 5 5 5 5 1 1 5 5 1 2 2 2 2 2 2 2 2 2 5 2 2 5 2 2 4 4 4
## [71] 4 4 4 4 4 4 4
##
## Within cluster sum of squares by cluster:
## [1] 56.89736 39.23482 26.74809 24.97754 38.49080
##  (between_SS / total_SS =  75.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
kmeans15.df$cluster
```

```
##  [1] 1 1 1 1 5 1 1 1 1 1 1 3 1 1 1 5 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3
## [36] 3 1 5 5 5 5 5 5 5 5 5 5 5 1 1 5 5 1 2 2 2 2 2 2 2 2 2 5 2 2 5 2 2 4 4 4
## [71] 4 4 4 4 4 4 4
```

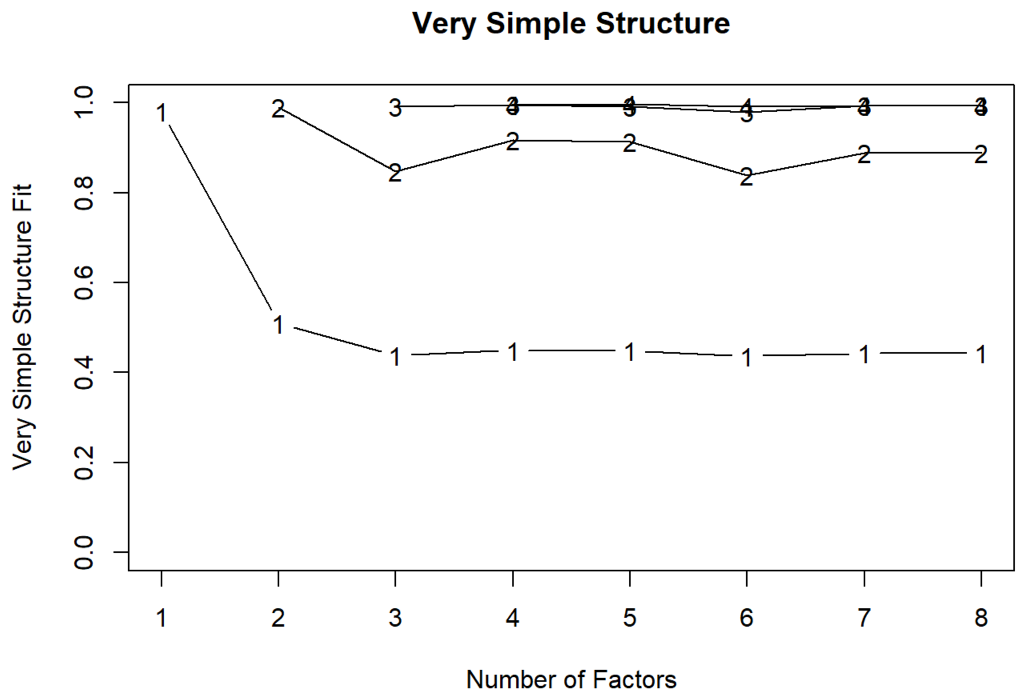
```
#If we look at the above results we can see that for maximum number of times we get cluster 1 with cluster 5
for datapoints so we can conclude that IndianWolves are related to Modern Dogs


#Converting gender back to factor variable
df$Gender <- as.factor(df$Gender)

#Question 5
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.5.3
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
vss(df[,-c(1,11)]) # See factor recommendation from Vss the recommended factor is 2
```




```
##
## Very Simple Structure
## Call: vss(x = df[, -c(1, 11)])
## VSS complexity 1 achieves a maximimum of 0.98  with  1  factors
## VSS complexity 2 achieves a maximimum of 0.99  with  2  factors
##
## The Velicer MAP achieves a minimum of 0.08  with  2  factors
## BIC achieves a minimum of  NA  with  2  factors
## Sample Size adjusted BIC achieves a minimum of  NA  with  4  factors
##
## Statistics by number of factors
##   vss1 vss2   map dof   chisq    prob sqresid  fit RMSEA BIC  SABIC
## 1 0.98 0.00 0.094  27 1.4e+02 2.6e-17    0.98 0.98 0.244  23 108.37
## 2 0.51 0.99 0.080  19 4.4e+01 8.6e-04    0.51 0.99 0.141 -38  21.67
## 3 0.44 0.85 0.121  12 1.5e+01 2.2e-01    0.38 0.99 0.072 -37   1.12
## 4 0.45 0.92 0.165   6 4.1e+00 6.6e-01    0.23 1.00 0.000 -22  -3.05
## 5 0.45 0.91 0.174   1 3.0e-01 5.8e-01    0.16 1.00 0.000  -4  -0.89
## 6 0.44 0.84 0.288  -3 4.7e-08      NA    0.13 1.00    NA  NA     NA
## 7 0.44 0.89 0.550  -6 1.4e-06      NA    0.28 0.99    NA  NA     NA
## 8 0.44 0.89 1.000  -8 0.0e+00      NA    0.28 0.99    NA  NA     NA
##   complex  eChisq    SRMR  eCRMS   eBIC
## 1     1.0 1.7e+01 5.5e-02 0.0635 -100.5
## 2     1.7 2.1e+00 2.0e-02 0.0269  -80.4
## 3     2.1 4.1e-01 8.6e-03 0.0150  -51.7
## 4     2.0 9.5e-02 4.1e-03 0.0101  -26.0
## 5     2.0 5.3e-03 9.8e-04 0.0059   -4.3
## 6     2.2 8.6e-10 3.9e-07     NA     NA
## 7     2.2 1.4e-08 1.6e-06     NA     NA
## 8     2.2 9.9e-18 4.2e-11     NA     NA
```

```
pc <- principal(df[,-c(1,11)], nfactors=2, rotate="varimax")
summary(pc)
```

```
##
## Factor analysis with Call: principal(r = df[, -c(1, 11)], nfactors = 2, rotate = "varimax")
##
## Test of the hypothesis that 2 factors are sufficient.
## The degrees of freedom for the model is 19  and the objective function was  1.18
## The number of observations was  77  with Chi Square =  83.28  with prob <  5e-10
##
## The root mean square of the residuals (RMSA) is  0.04
```

```
round(pc$values, 3)
```

```
## [1] 7.052 0.704 0.543 0.193 0.180 0.132 0.092 0.070 0.035
```

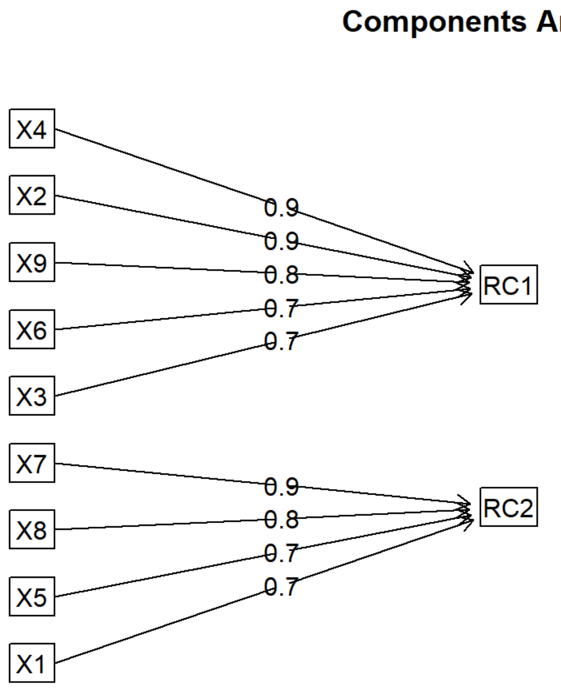
```
pc$loadings
```

```
##
## Loadings:
##    RC1   RC2
## X1 0.672 0.700
## X2 0.863 0.390
## X3 0.713 0.255
## X4 0.903 0.278
## X5 0.638 0.701
## X6 0.736 0.553
## X7 0.203 0.932
## X8 0.544 0.783
## X9 0.756 0.564
##
##                  RC1   RC2
## SS loadings    4.375 3.381
## Proportion Var 0.486 0.376
## Cumulative Var 0.486 0.862
```

```
#From the loadings we can see that upto 2 RC factors explain about 86% of the variance.

pc$communality
```

```
##        X1        X2        X3        X4        X5        X6        X7
## 0.9417787 0.8968812 0.5732902 0.8922558 0.8976670 0.8472377 0.9093146
##        X8        X9
## 0.9082632 0.8897015
```

```
#Plotting the EFA Plot
fa.diagram(pc)
```




```
#Question 6- Discriminant Function Analysis

library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.5.2
```

```
library(klaR)
```

```
## Warning: package 'klaR' was built under R version 3.5.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

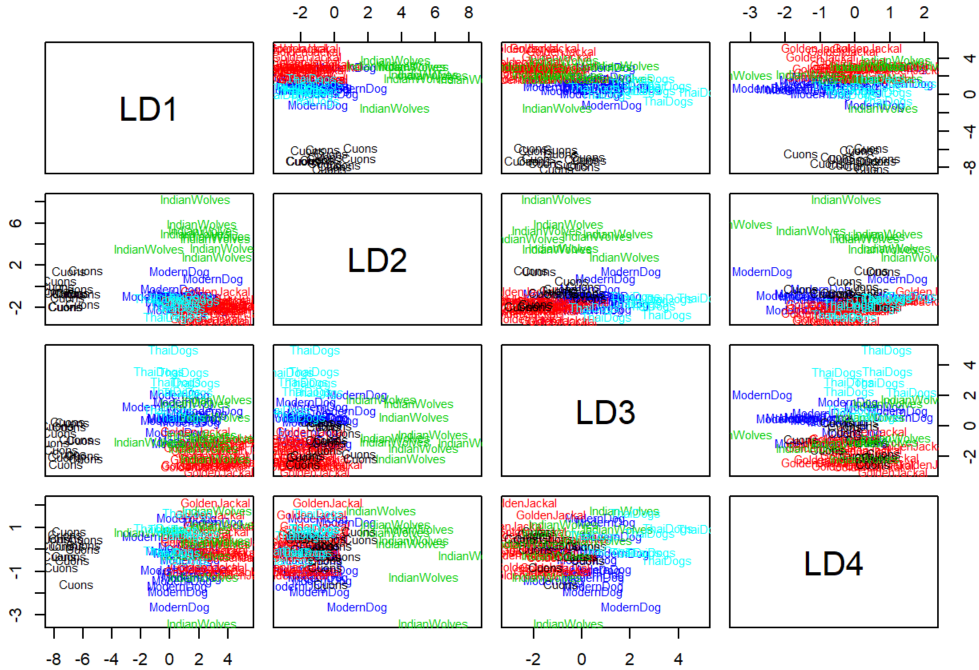
```
new_data <- df[,-11]
sample_n(new_data, 10)
```

```
##     CanineGroup  X1   X2 X3 X4 X5  X6 X7 X8  X9
## 1  GoldenJackal 110  7.3 19 15 17 6.1 30 33 4.5
## 2  IndianWolves 164 10.7 24 23 26 9.5 43 47 7.6
## 3     ModernDog 121 10.2 18 21 21 7.9 35 38 6.2
## 4         Cuons 123  9.7 22 21 20 7.8 27 36 6.1
## 5         Cuons 131 10.9 25 24 21 8.5 29 35 6.2
## 6         Cuons 144 10.8 24 26 22 8.9 30 42 7.1
## 7  GoldenJackal 107  8.4 18 17 18 6.2 29 31 4.3
## 8  IndianWolves 164 12.1 27 24 25 9.9 42 45 8.3
## 9         Cuons 138 11.4 25 25 22 9.0 30 38 7.3
## 10 GoldenJackal 111  8.5 19 16 18 7.1 30 33 5.0
```

```
training_sample <- sample(c(TRUE, FALSE), nrow(new_data), replace = T, prob =
                            c(0.8,0.2))
train <- new_data[training_sample, ]
test <- new_data[!training_sample, ]
lda.new_data <- lda(CanineGroup ~ ., train)
lda.new_data #show results
```

```
## Call:
## lda(CanineGroup ~ ., data = train)
##
## Prior probabilities of groups:
##        Cuons GoldenJackal IndianWolves    ModernDog     ThaiDogs
##    0.2089552    0.2985075    0.1343284    0.2238806    0.1343284
##
## Group means:
##                     X1       X2       X3       X4       X5       X6
## Cuons         133.5714 10.77857 22.71429 23.71429 21.28571 8.485714
## GoldenJackal 111.0000  8.18000 18.60000 17.00000 18.20000 6.815000
## IndianWolves 153.1111 11.34444 25.33333 24.22222 24.33333 9.144444
## ModernDog    126.0000  9.74000 21.26667 21.20000 19.33333 7.666667
## ThaiDogs     121.3333 10.16667 19.77778 22.66667 19.11111 8.155556
##                     X7       X8       X9
## Cuons         28.78571 37.78571 6.585714
## GoldenJackal 30.35000 33.35000 4.805000
## IndianWolves 39.44444 44.00000 7.266667
## ModernDog    32.00000 36.60000 5.840000
## ThaiDogs     32.44444 35.55556 6.077778
##
## Coefficients of linear discriminants:
##            LD1         LD2         LD3         LD4
## X1 -0.05020915  0.02560711 -0.09156036 -0.13150599
## X2  0.05304885 -0.01646892  0.65437137 -0.51168290
## X3  0.07232645  0.04135098 -0.02636617 -0.03238414
## X4 -0.22417983 -0.09464178  0.39475755  0.12850194
## X5 -0.68463926  0.87647018 -1.24862354  0.63044480
## X6 -0.97143978 -0.29880051  0.39143201  1.16240865
## X7  1.23799377  0.19354480  0.39755147  0.04345402
## X8  0.10066084  0.08935177  0.08074078 -0.01974962
## X9 -1.50482810 -0.50763822  1.16117709 -0.14651620
##
## Proportion of trace:
##    LD1    LD2    LD3    LD4
## 0.6688 0.2264 0.0990 0.0057
```

```
plot(lda.new_data, col = as.integer(train$CanineGroup))
```




```
lda.train <- predict(lda.new_data)
train$lda <- lda.train$class
table(train$lda,train$CanineGroup)
```

```
##
##                Cuons GoldenJackal IndianWolves ModernDog ThaiDogs
##   Cuons           14            0            0         0        0
##   GoldenJackal     0           19            0         0        0
##   IndianWolves     0            0            9         0        0
##   ModernDog        0            1            0        14        2
##   ThaiDogs         0            0            0         1        7
```

```
lda.test <- predict(lda.new_data,test)
test$lda <- lda.test$class
table(test$lda,test$CanineGroup)
```

```
##
##                Cuons GoldenJackal IndianWolves ModernDog ThaiDogs
##   Cuons           3            0            0         0        0
##   GoldenJackal    0            0            0         0        0
##   IndianWolves    0            0            5         0        0
##   ModernDog       0            0            0         1        0
##   ThaiDogs        0            0            0         0        1
```

```
#Question7: logistic regression for each Canine group

#For Cuons

#Extracting dfata for cuons
df.cuons <- df[df$CanineGroup=='Cuons',]
df.cuons <- df.cuons[,-1]
#Changing the levels to 0&1 i.e Females and Males
levels(df.cuons$Gender) <- c(0,1,1)

#Applying Logistic regression to with all the variables for Cuons
fit.cuons <- glm(Gender~.,data=df.cuons,family = 'binomial')
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#Applying stepwise regression to find best variables
final.fit.cuons <- step(fit.cuons)
```

```
## Start:  AIC=20
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##         Df   Deviance AIC
## - X2     1 9.3930e-10  18
## - X7     1 9.6750e-10  18
## - X5     1 9.9310e-10  18
## - X4     1 9.9390e-10  18
## - X1     1 1.0473e-09  18
## - X8     1 1.0549e-09  18
## - X3     1 2.0975e-09  18
## - X6     1 5.3654e-09  18
## - X9     1 7.2480e-09  18
## <none>     9.3450e-10  20
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=18
## Gender ~ X1 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##         Df   Deviance AIC
## - X7     1 1.0180e-09  16
## - X4     1 1.0206e-09  16
## - X5     1 1.1232e-09  16
## - X1     1 1.4116e-09  16
## - X8     1 1.5474e-09  16
## - X3     1 4.3633e-09  16
## - X9     1 6.9469e-09  16
## - X6     1 1.2524e-08  16
## <none>     9.3930e-10  18
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=16
## Gender ~ X1 + X3 + X4 + X5 + X6 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X4    1    0.000 14.000
## - X1    1    0.000 14.000
## - X8    1    0.000 14.000
## - X5    1    0.000 14.000
## - X3    1    0.000 14.000
## - X9    1    0.000 14.000
## <none>       0.000 16.000
## - X6    1   10.866 24.866
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=14
## Gender ~ X1 + X3 + X5 + X6 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X1    1   0.0000 12.000
## - X8    1   0.0000 12.000
## - X5    1   0.0000 12.000
## <none>      0.0000 14.000
## - X3    1   7.7103 19.710
## - X9    1  16.6998 28.700
## - X6    1  18.1660 30.166
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=12
## Gender ~ X3 + X5 + X6 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X8    1   0.0000 10.000
## - X5    1   0.0000 10.000
## <none>      0.0000 12.000
## - X3    1   7.8986 17.899
## - X9    1  17.6290 27.629
## - X6    1  18.1660 28.166
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=10
## Gender ~ X3 + X5 + X6 + X9
##
##         Df Deviance    AIC
## <none>       0.0000 10.000
## - X5     1   7.8475 15.848
## - X3     1   8.0938 16.094
## - X9     1  18.1685 26.169
## - X6     1  18.5302 26.530
```

```
#For cuons we get X3,X5,X6,X9 as most contributing variables to predict gender

#For ModernDog

#Extracting dfata for cuons
df.moderndog <- df[df$CanineGroup=='ModernDog',]
df.moderndog <- df.moderndog[,-1]
#Changing the levels to 0&1 i.e Females and Males
levels(df.moderndog$Gender) <- c(0,1,1)

#Applying Logistic regression to with all the variables for ModernDog
fit.moderndog <- glm(Gender~.,data=df.moderndog,family = 'binomial')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#Applying stepwise regression to find best variables
final.fit.moderndog <- step(fit.moderndog)
```

```
## Start:  AIC=20
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##          Df   Deviance AIC
## - X2      1 5.2400e-10  18
## - X3      1 5.4300e-10  18
## - X6      1 5.6300e-10  18
## - X5      1 5.7400e-10  18
## - X9      1 6.9800e-10  18
## - X7      1 9.3200e-10  18
## - X4      1 1.0200e-09  18
## - X8      1 7.3050e-09  18
## - X1      1 3.8625e-08  18
## <none>      4.3800e-10  20
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=18
## Gender ~ X1 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##          Df Deviance   AIC
## - X3      1   0.0000 16.00
## - X6      1   0.0000 16.00
## - X9      1   0.0000 16.00
## - X5      1   0.0000 16.00
## - X7      1   0.0000 16.00
## - X4      1   0.0000 16.00
## - X8      1   0.0000 16.00
## <none>        0.0000 18.00
## - X1      1   9.8201 25.82
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=16
## Gender ~ X1 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X6    1   0.0000 14.000
## - X5    1   0.0000 14.000
## - X9    1   0.0000 14.000
## - X4    1   0.0000 14.000
## - X7    1   0.0000 14.000
## <none>      0.0000 16.000
## - X8    1   8.8363 22.836
## - X1    1  10.2749 24.275
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=14
## Gender ~ X1 + X4 + X5 + X7 + X8 + X9
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X5    1   0.0000 12.000
## - X9    1   0.0000 12.000
## <none>      0.0000 14.000
## - X4    1   7.1863 19.186
## - X8    1   9.2476 21.248
## - X1    1  11.1937 23.194
## - X7    1  14.5018 26.502
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=12
## Gender ~ X1 + X4 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X9    1   0.0000 10.000
## <none>      0.0000 12.000
## - X4    1   9.4321 19.432
## - X8    1   9.4832 19.483
## - X1    1  11.3406 21.341
## - X7    1  16.3343 26.334
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=10
## Gender ~ X1 + X4 + X7 + X8
##
##        Df Deviance    AIC
## <none>      0.0000 10.000
## - X4    1   9.9914 17.991
## - X8    1  11.2905 19.291
## - X1    1  12.8696 20.870
## - X7    1  18.1140 26.114
```

```
#For ModernDog we get variables X4,X8,X1,X7 as most contributing variables

#For GoldenJackal

#Extracting dfata for cuons
df.GoldenJackal <- df[df$CanineGroup=='GoldenJackal',]
df.GoldenJackal <- df.GoldenJackal[,-1]
#Changing the levels to 0&1 i.e Females and Males
levels(df.GoldenJackal$Gender) <- c(0,1,1)

#Applying Logistic regression to with all the variables for GoldenJackal
fit.GoldenJackal <- glm(Gender~.,data=df.GoldenJackal,family = 'binomial')
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#Applying stepwise regression to find best variables
final.fit.GoldenJackal <- step(fit.GoldenJackal)
```

```
## Start:  AIC=20
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X3    1 1.1501e-09  18
## - X8    1 1.1747e-09  18
## - X4    1 1.1758e-09  18
## - X9    1 1.3420e-09  18
## - X5    1 1.3552e-09  18
## - X6    1 1.5406e-09  18
## - X1    1 1.7927e-09  18
## - X7    1 1.8378e-09  18
## - X2    1 4.2338e-09  18
## <none>    1.1415e-09  20
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=18
## Gender ~ X1 + X2 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X4    1 1.2020e-09  16
## - X8    1 1.4551e-09  16
## - X6    1 1.5372e-09  16
## - X5    1 1.8172e-09  16
## - X7    1 1.8482e-09  16
## - X1    1 2.6216e-09  16
## - X9    1 3.0477e-09  16
## - X2    1 3.8965e-09  16
## <none>    1.1501e-09  18
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=16
## Gender ~ X1 + X2 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X8    1 1.4356e-09  14
## - X7    1 1.7422e-09  14
## - X5    1 2.3021e-09  14
## - X9    1 3.6386e-09  14
## - X6    1 4.0799e-09  14
## - X1    1 1.0388e-08  14
## - X2    1 1.2225e-08  14
## <none>    1.2020e-09  16
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=14
## Gender ~ X1 + X2 + X5 + X6 + X7 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X7    1   0.0000 12.000
## - X5    1   0.0000 12.000
## - X9    1   0.0000 12.000
## - X2    1   0.0000 12.000
## - X1    1   0.0000 12.000
## <none>      0.0000 14.000
## - X6    1   5.6182 17.618
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=12
## Gender ~ X1 + X2 + X5 + X6 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X5    1   0.0000 10.000
## - X9    1   0.0000 10.000
## - X2    1   0.0000 10.000
## - X1    1   0.0000 10.000
## <none>      0.0000 12.000
## - X6    1   5.6261 15.626
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=10
## Gender ~ X1 + X2 + X6 + X9
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X2    1   0.0000  8.000
## - X1    1   0.0000  8.000
## <none>      0.0000 10.000
## - X9    1   7.7888 15.789
## - X6    1   8.4675 16.468
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=8
## Gender ~ X1 + X6 + X9
##
##        Df Deviance    AIC
## <none>      0.0000  8.000
## - X1    1   7.8698 13.870
## - X6    1   8.6179 14.618
## - X9    1   9.8447 15.845
```

```
#From the above results we can say that for GoldenJackal we have X1,X6,X9 as he most contributing variables

#For IndianWolves

#Extracting dfata for cuons
df.IndianWolves <- df[df$CanineGroup=='IndianWolves',]
df.IndianWolves <- df.IndianWolves[,-1]
#Changing the levels to 0&1 i.e Females and Males
levels(df.IndianWolves$Gender) <- c(0,1,1)

#Applying Logistic regression to with all the variables for IndianWolves
fit.IndianWolves <- glm(Gender~.,data=df.IndianWolves,family = 'binomial')
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#Applying stepwise regression to find best variables
final.fit.IndianWolves <- step(fit.IndianWolves)
```

```
## Start:  AIC=20
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X9    1 2.3819e-10  18
## - X2    1 2.3871e-10  18
## - X4    1 3.8639e-10  18
## - X6    1 4.8390e-10  18
## - X3    1 5.0125e-10  18
## - X1    1 5.0833e-10  18
## - X5    1 5.3413e-10  18
## - X8    1 5.4460e-10  18
## - X7    1 5.7355e-10  18
## <none>    3.4651e-10  20
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=18
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X4    1 2.7584e-10  16
## - X8    1 4.2524e-10  16
## - X6    1 4.4206e-10  16
## - X1    1 4.8572e-10  16
## - X3    1 5.1832e-10  16
## - X7    1 6.2639e-10  16
## - X5    1 7.1771e-10  16
## - X2    1 1.6137e-09  16
## <none>    2.3819e-10  18
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=16
## Gender ~ X1 + X2 + X3 + X5 + X6 + X7 + X8
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X6    1 4.9580e-10  14
## - X8    1 6.0980e-10  14
## - X3    1 6.9290e-10  14
## - X1    1 6.9320e-10  14
## - X7    1 8.1330e-10  14
## - X5    1 1.5389e-09  14
## - X2    1 3.4172e-09  14
## <none>    2.7580e-10  16
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=14
## Gender ~ X1 + X2 + X3 + X5 + X7 + X8
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df   Deviance AIC
## - X8    1 6.8620e-10  12
## - X3    1 7.5510e-10  12
## - X1    1 7.6540e-10  12
## - X7    1 8.8060e-10  12
## - X5    1 1.7412e-09  12
## - X2    1 5.0559e-09  12
## <none>    4.9580e-10  14
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=12
## Gender ~ X1 + X2 + X3 + X5 + X7
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X1    1    0.000 10.000
## - X3    1    0.000 10.000
## - X7    1    0.000 10.000
## - X5    1    0.000 10.000
## <none>       0.000 12.000
## - X2    1   10.146 20.146
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=10
## Gender ~ X2 + X3 + X5 + X7
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##        Df Deviance    AIC
## - X3    1    0.000  8.000
## - X7    1    0.000  8.000
## - X5    1    0.000  8.000
## <none>       0.000 10.000
## - X2    1   10.727 18.727
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=8
## Gender ~ X2 + X5 + X7
##
##        Df Deviance    AIC
## <none>      0.0000  8.000
## - X7    1   5.3755 11.376
## - X5    1   8.0172 14.017
## - X2    1  11.7552 17.755
```

```
#For IndianWolves we get most contributing factors as X7,X5,X2.

#If we analyze the above results we get most frequent variables as X1,X5,X6,X7,X9 as the contributing variabl
es

#Hence we can fit a logistic regression with variables X1,X5,X6,X7,X9,X2


#Verifying the above results
df.all.except.thaidogs <- df[!(df$CanineGroup =='ThaiDogs'),]
df.all.except.thaidogs <- df.all.except.thaidogs[,-1]
#Changing the levels to 0&1 i.e Females and Males
levels(df.all.except.thaidogs$Gender) <- c(0,1,1)

#Applying Logistic regression to with all the variables for ThaiDogs
fit.all.except.thaidogs <- glm(Gender~.,data=df.all.except.thaidogs,family = 'binomial')
#Applying stepwise regression to find best variables
final.fit.all.except.thaidogs <- step(fit.all.except.thaidogs)
```

```
## Start:  AIC=101.94
## Gender ~ X1 + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9
##
##        Df Deviance    AIC
## - X5    1   82.144 100.14
## - X7    1   82.258 100.26
## - X1    1   82.452 100.45
## - X6    1   82.741 100.74
## - X8    1   83.076 101.08
## - X4    1   83.082 101.08
## - X3    1   83.130 101.13
## - X2    1   83.696 101.70
## - X9    1   83.781 101.78
## <none>      81.939 101.94
##
## Step:  AIC=100.14
## Gender ~ X1 + X2 + X3 + X4 + X6 + X7 + X8 + X9
##
##        Df Deviance     AIC
## - X7    1   82.336  98.336
## - X1    1   82.682  98.682
## - X4    1   83.308  99.308
## - X3    1   83.322  99.322
## - X8    1   83.386  99.386
## - X6    1   83.579  99.579
## - X9    1   83.798  99.798
```

```
## - X2    1   83.837  99.837
## <none>      82.144 100.144
##
## Step:  AIC=98.34
## Gender ~ X1 + X2 + X3 + X4 + X6 + X8 + X9
##
##        Df Deviance    AIC
## - X1    1   83.049 97.049
## - X8    1   83.398 97.398
## - X3    1   83.564 97.564
## - X6    1   83.603 97.603
## - X9    1   83.856 97.856
## - X2    1   84.170 98.170
## <none>      82.336 98.336
## - X4    1   84.480 98.480
##
## Step:  AIC=97.05
## Gender ~ X2 + X3 + X4 + X6 + X8 + X9
##
##        Df Deviance    AIC
## - X8    1   83.400 95.400
## - X6    1   83.919 95.919
## - X9    1   84.563 96.563
## - X4    1   84.719 96.719
## - X2    1   84.871 96.871
## - X3    1   84.937 96.937
## <none>      83.049 97.049
##
## Step:  AIC=95.4
## Gender ~ X2 + X3 + X4 + X6 + X9
##
##        Df Deviance    AIC
## - X9    1   84.578 94.578
## - X6    1   84.634 94.634
## - X3    1   85.094 95.094
## - X4    1   85.171 95.171
## <none>      83.400 95.400
## - X2    1   85.456 95.456
##
## Step:  AIC=94.58
## Gender ~ X2 + X3 + X4 + X6
##
##        Df Deviance    AIC
## - X6    1   84.936 92.936
## - X3    1   86.121 94.121
## - X4    1   86.297 94.297
## <none>      84.578 94.578
## - X2    1   88.241 96.241
##
## Step:  AIC=92.94
## Gender ~ X2 + X3 + X4
##
##        Df Deviance    AIC
## - X3    1   86.415 92.415
## <none>      84.936 92.936
## - X4    1   87.201 93.201
## - X2    1   88.358 94.358
##
## Step:  AIC=92.41
## Gender ~ X2 + X4
##
##        Df Deviance    AIC
## - X4    1   87.911 91.911
## <none>      86.415 92.415
## - X2    1   90.416 94.416
##
## Step:  AIC=91.91
## Gender ~ X2
##
##        Df Deviance    AIC
## <none>      87.911 91.911
## - X2    1   92.747 94.747
```

```r
#From above results we get that X2 is the contributing variable. Thus we fit the new model with minimum numb
er of parameters

#Question9:

#Training the model with entire data and parameters X1,X5,X6,X7,X9,X2
df.all.except.thaidogs <- df[!(df$CanineGroup =='ThaiDogs'),]
df.all.except.thaidogs <- df.all.except.thaidogs[,-1]

#Extracting the relevant columns
df.all.except.thaidogs <-   df.all.except.thaidogs[,c('X1','X5','X6','X7','X9','X2','Gender')]

#Changing the levels to 0&1 i.e Females and Males
levels(df.all.except.thaidogs$Gender) <- c(0,1,1)
#Applying Logistic regression to with all the variables
fit.all.except.thaidogs <- glm(Gender~.,data=df.all.except.thaidogs,family ='binomial')
#Calculating the accuracy of Logistic Regression
cm <-table(df.all.except.thaidogs$Gender,as.factor(ifelse(test=as.numeric(fit.all.except.thaidogs$fitted.val
ues>0.5) == 0, yes=1, no=0)))
accuracy <- (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
accuracy
```

```
## [1] 0.4328358
```

```r
library(pROC)
```

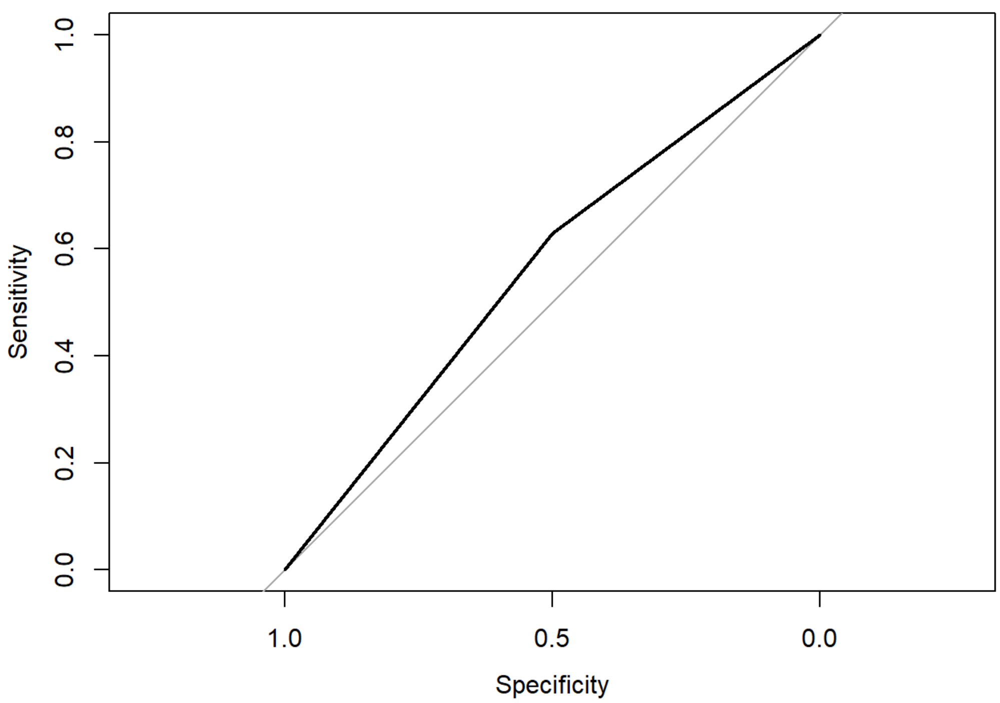
```
## Warning: package 'pROC' was built under R version 3.5.2
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
roc(df.all.except.thaidogs$Gender,ifelse(test=as.numeric(fit.all.except.thaidogs$fitted.values>0.5) == 0, ye
s=1, no=0),plot=TRUE)
```



```
##
## Call:
## roc.default(response = df.all.except.thaidogs$Gender, predictor = ifelse(test = as.numeric(fit.all.except.thaidogs$fitted.values >     0.5) == 0, yes = 1, no = 0), plot = TRUE)
##
## Data: ifelse(test = as.numeric(fit.all.except.thaidogs$fitted.values > 0.5) == 0, yes = 1, no = 0) in 32 controls (df.all.except.thaidogs$Gender 0) > 35 cases (df.all.except.thaidogs$Gender 1).
## Area under the curve: 0.5643
```

```
#Question 8 answer: Area under curve is very less hence, the model has low accuracy.
#Ans for question 9a: We used Logistic Regression becsause we had binary outcome so Logistic
#Ans for question 9b: We get an accuracy of 43.28% for Logistic Regression


#Creating test data
df.test <- df[(df$CanineGroup =='ThaiDogs'),c('X1','X5','X6','X7','X9','X2','Gender')]

#Predicting the values
predicted_values <- predict(fit.all.except.thaidogs,newdata = df.test[,-c(7)])

#Predicting the values of Male & Female
predicted_values <- as.factor(ifelse(test=as.numeric(predicted_values>0.5) == 0, yes="Male", no="Female"))

predicted_values
```

```
##  [1] Female Female Female Male   Female Female Male   Female Female Male
## Levels: Female Male
```

```
df[df$CanineGroup=='ThaiDogs',]$Gender<-as.factor(ifelse(test=as.numeric(predicted_values>0.5) == 0, yes=2, no=1))
```

```
## Warning in Ops.factor(predicted_values, 0.5): '>' not meaningful for
## factors
```

```
#Answer10: We have to create a linear regression model to predict Mandible Length i.e X1

#Extracting all the data except for Thaidogs
names(df)
```

```
##  [1] "CanineGroup" "X1"          "X2"          "X3"          "X4"
##  [6] "X5"          "X6"          "X7"          "X8"          "X9"
## [11] "Gender"
```

```
df.all.except.thaidogs <- df[!(df$CanineGroup =='ThaiDogs'),c(2:10)]

#Creating the model with X1 against all the variables
fit.lm <- lm(X1~.,data = df.all.except.thaidogs)
summary(fit.lm)
```

```
##
## Call:
## lm(formula = X1 ~ ., data = df.all.except.thaidogs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0282  -1.9666   0.2624   2.6000  10.2278
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -12.6009     4.9450  -2.548  0.01350 *
## X2            0.4696     1.0314   0.455  0.65061
## X3            0.3451     0.1542   2.238  0.02911 *
## X4            1.1334     0.4196   2.701  0.00904 **
## X5            0.3042     0.6434   0.473  0.63813
## X6            0.1986     1.3123   0.151  0.88022
## X7            0.4763     0.2299   2.072  0.04273 *
## X8            1.9582     0.2897   6.759 7.44e-09 ***
## X9            1.5098     1.4110   1.070  0.28905
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.979 on 58 degrees of freedom
## Multiple R-squared:  0.9586, Adjusted R-squared:  0.9529
## F-statistic:   168 on 8 and 58 DF,  p-value: < 2.2e-16
```

```
#From the above summary,if we look at adjusted R-squared value, we can conclude that the accuracy is around 95%

#Now we predict the values for Thai Dogs
#Creating test data
df.test.thaidogs <- df[(df$CanineGroup =='ThaiDogs'),c(2:10)]
#Predicting the values for X1
predicted_values <- predict(fit.lm,newdata =df.test.thaidogs[,-c(1)])

#Loading the required library
library(ggplot2)
#Plotting the predicted values with actual values
qplot(df.test.thaidogs$X1, predicted_values) + geom_abline(intercept=0,slope=1)
```