Brand Maps for Automobiles

Objective : Compare Different Car Brands using Principal Component Analysis and Provide recomendations to Infinity Brand to improve performance

Loading Libraries and Dataset

```
### Clear WorkSpace
rm(list=ls(all=TRUE))

### Loading Packages
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-2

library(readxl)

### Loading Data
data <- read_excel("Cars_Data.xlsx",sheet=1)    # read csv file and label the
data as "data"
y <-  as.matrix(data[,17])
x <-  as.matrix(data[,2:16])
```

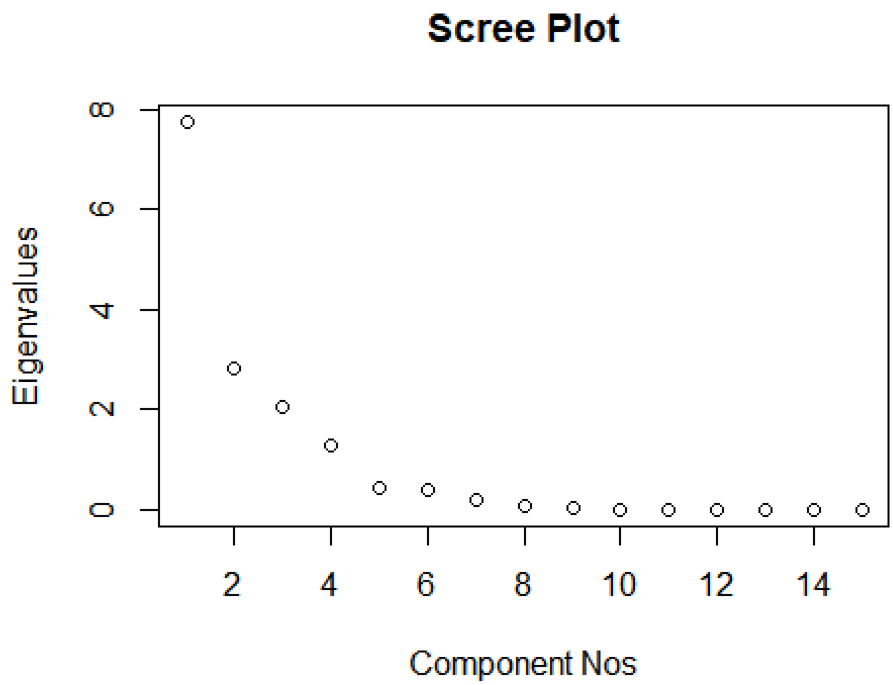
Variable Ratings Available for Analysis

```
colnames(x)

##  [1] "Attractive"    "Quiet"          "Unreliable"    "Poorly Built"
##  [5] "Interesting"   "Sporty"         "Uncomfortable" "Roomy"
##  [9] "Easy Service"  "Prestige"       "Common"        "Economical"
## [13] "Successful"    "AvantGarde"     "Poor Value"
```

Creating Eigen Vector and Eigen Values

```
### correlation matrix of available variables
cor_mat = cor(x)

### eigen decomposition of correlation matrix
output1 <-  eigen(cor_mat)

### eigenvalues
va <-  output1$values

### eigenvector
ve <-  output1$vectors

### Plotting Eigen Values
```

```
### scree plot
plot(va, ylab = "Eigenvalues", xlab = "Component Nos", main="Scree Plot")
```



Retaining Eigen Values > 1 and Making Smaller Values < 0.3 to 0

```
### eigenvalues > 1
ego <- va[va > 1]

### number of factors to retain
nn <- nrow(as.matrix(ego))

### eigenvectors associated with the reatined factors
### ignore small values < 0.3
out2 <- ve[,1:nn]
out3 <- ifelse(abs(out2) < 0.3, 0, out2)

rownames(out3) <- c("Attractive","Quiet","Unreliable","Poorly
Built","Interesting","Sporty","Uncomfortable","Roomy","Easy
Service","Prestige","Common","Economical","Successful","AvantGarde","Poor
Value" )
```

Building Principle Component Regression

```
### Component Scores; coordinates of the brands in the map
z = x %*% out3

### Preference Regression to estimate how benefits drive overall preferences
```

```r
= f(benefits)
out5 = lm(y ~ z)

### Summary
summary(out5)

##
## Call:
## lm(formula = y ~ z)
##
## Residuals:
##        1        2        3        4        5        6        7        8
## -0.41399  0.20488  0.21196  0.17883 -0.17872  0.29476  0.07001 -0.13008
##        9       10
## -0.11977 -0.11788
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.18458    0.43161   5.061 0.003895 **
## z1          -0.68944    0.07258  -9.499 0.000219 ***
## z2          -0.28516    0.11920  -2.392 0.062209 .
## z3           0.46138    0.11900   3.877 0.011675 *
## z4           0.21219    0.21267   0.998 0.364205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.303 on 5 degrees of freedom
## Multiple R-squared:  0.9742, Adjusted R-squared:  0.9536
## F-statistic: 47.21 on 4 and 5 DF,  p-value: 0.0003671
```

The components z1 and z2 are negative. Lets adjust them to positive to make people understand scale properly

```r
### Flipping Signs
out3[,1] = (-1)*out3[,1]
out3[,2] = (-1)*out3[,2]
out3

##                     [,1]       [,2]       [,3]       [,4]
## Attractive     0.3262814  0.0000000  0.0000000  0.0000000
## Quiet          0.3173818  0.0000000  0.0000000  0.0000000
## Unreliable     0.0000000  0.4011014  0.0000000  0.0000000
## Poorly Built  -0.3319244  0.0000000  0.0000000  0.0000000
## Interesting    0.0000000  0.0000000 -0.4290683  0.0000000
## Sporty         0.0000000 -0.4288285  0.0000000  0.0000000
## Uncomfortable  0.0000000  0.0000000  0.0000000  0.0000000
## Roomy          0.0000000  0.3789204  0.0000000  0.0000000
## Easy Service   0.0000000 -0.4279673  0.0000000  0.0000000
## Prestige       0.3322855  0.0000000  0.0000000  0.0000000
## Common         0.0000000  0.0000000  0.0000000 -0.4663407
## Economical     0.0000000  0.0000000  0.6642165  0.0000000
```

```
## Successful     0.3155394  0.0000000  0.0000000  0.0000000
## AvantGarde     0.0000000  0.0000000  0.0000000  0.6685725
## Poor Value     0.0000000  0.0000000 -0.4710589  0.0000000
```

Defining Principle Components based on variance contribution

Z1 = Premium

Z2 = Heavy

Z3 = Economy

Z4 = Unique

```
### Principle Component Regression
z = x %*% out3

### Preference Regression to estimate how benefits drive overall preferences
= f(benefits)
out5 = lm(y ~ z)

### Summary
summary(out5)

##
## Call:
## lm(formula = y ~ z)
##
## Residuals:
##        1        2        3        4        5        6        7        8
## -0.41399  0.20488  0.21196  0.17883 -0.17872  0.29476  0.07001 -0.13008
##        9       10
## -0.11977 -0.11788
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.18458    0.43161   5.061 0.003895 **
## z1           0.68944    0.07258   9.499 0.000219 ***
## z2           0.28516    0.11920   2.392 0.062209 .
## z3           0.46138    0.11900   3.877 0.011675 *
## z4           0.21219    0.21267   0.998 0.364205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.303 on 5 degrees of freedom
## Multiple R-squared:  0.9742, Adjusted R-squared:  0.9536
## F-statistic: 47.21 on 4 and 5 DF,  p-value: 0.0003671
```

We will consider Z1 and Z3 as their coefficient is more compared to other components

Iso Preference Line and Ideal Vector

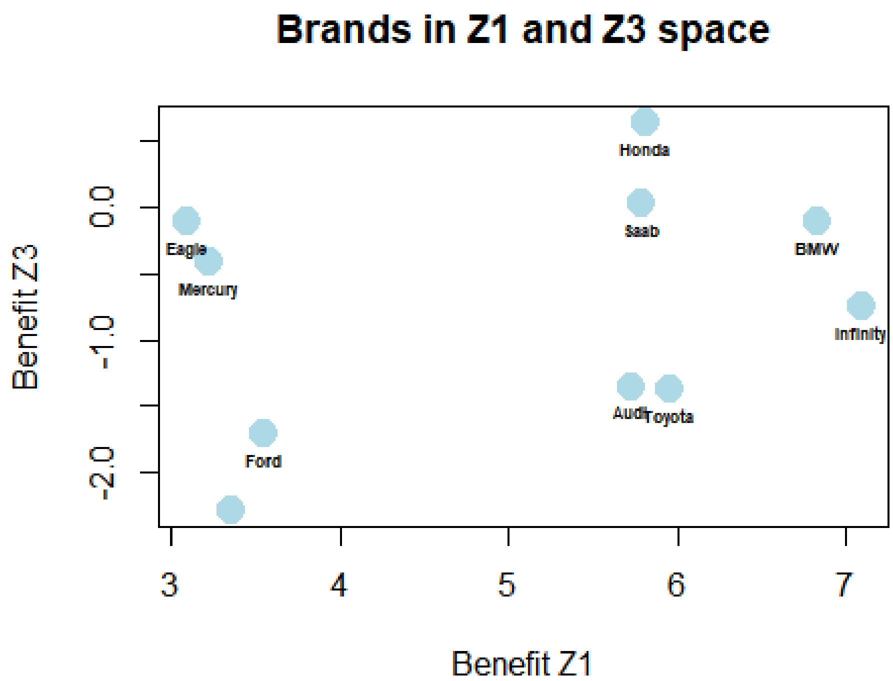
ISO Preference Line : Iso preference lines are composed of offers/brands that are equally preferred by a consumer. They are built by acquiring customer preferences for a brand or an offer.

Ideal vector : The ideal vector is a vector always perpendicular to the iso-preference line. It shows the direction in which the brand should be moved to improve the product design and increase its preference. The ideal vector is created by integrating all of the customers' preferences, allowing it to define the direction in which their preferences are growing.

```
## Consider factors z1 and z2 with positive slopes on preferences
## Let z2 be the y-axis and z1 as the x-axis
## Plot (z2, z1) to get brands in factor space of benefit 1 and benefit 2

### coordinates of brands in (z1, z2) space
Z1 = z[,1]
Z2 = z[,3]
z.out = cbind(Z1, Z2)
rownames(z.out) = data$Brands
data$Brands
```

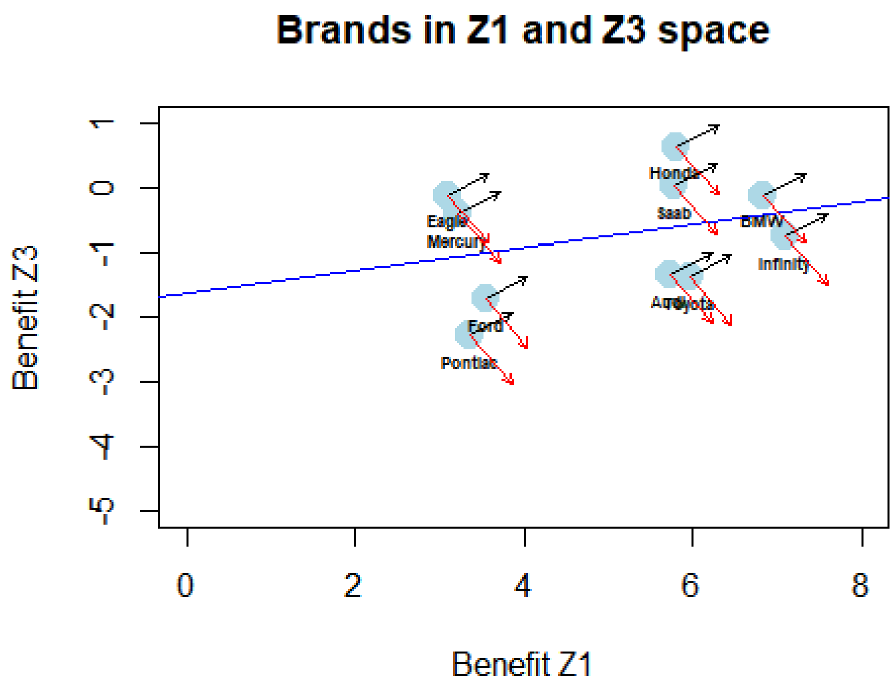
```
##  [1] "Infinity" "Ford"     "Audi"     "Toyota"   "Eagle"    "Honda"
##  [7] "Saab"     "Pontiac"  "BMW"      "Mercury"
```

```
### Plot, add labels, and save the brand map
plot(Z1, Z2, main = "Brands in Z1 and Z3 space", xlab = "Benefit Z1", ylab =
"Benefit Z3", col = "lightblue", pch = 19, cex = 2)# Brand Map in Z1-Z2 space
text(z.out, labels = row.names(z.out), font = 2, cex = 0.5, pos = 1)
# labeling brands
```




```
### Slopes of iso-preference and ideal vector
b1 = as.vector(coef(out5)[2])
b2 = as.vector(coef(out5)[4])
slope.iso.preference = - b1/b2
slope.iso.preference

## [1] -1.494305

slope.ideal.vector = b2/b1
slope.ideal.vector

## [1] 0.6692073

### Angles of iso-preference and ideal vector
angle.iso.preference = atan(slope.iso.preference)*180/pi
angle.iso.preference

## [1] -56.20927

angle.ideal.vector = atan(slope.ideal.vector)*180/pi
angle.ideal.vector

## [1] 33.79073

### Now do this for other pairs of significant benefit factors: (Z2, Z4), and
(Z4, Z1) b/c Z3 is not significant
plot(Z1, Z2, main = "Brands in Z1 and Z3 space", xlab = "Benefit Z1", ylab =
"Benefit Z3", col = "lightblue", pch = 19, cex = 2, xlim=c(0,8), ylim=c(-
```

```
5,1))# Brand Map in Z1-Z2 space
text(z.out, labels = row.names(z.out), font = 2, cex = 0.5, pos = 1)
abline(lm(Z2~Z1), col = "blue")
length <- 0.5
arrows(Z1, Z2, x1=Z1+length ,y1= slope.ideal.vector * (Z1+length) + Z2 -
slope.ideal.vector*Z1,
        length=0.05, col="Black")
arrows(Z1, Z2, x1=Z1+length ,y1= slope.iso.preference * (Z1+length) + Z2 -
slope.iso.preference*Z1,
       length=0.05, col="Red")
```



Recomendation for Infinity Brand

Based on the brand map, where we have Z1 (premium) and Z3 (economy) as the axes, we can see that for infinity, BMW is the closest competitor and is at a slightly higher iso-preference line for customers. In the case of Z1 (premium), infinity is the one with the highest score, so when it comes to premium cars, customers prefer infinity, which is not the case with Z3 (economy). In Z3, BMW has a much better score, and that is what infinity should focus on - making premium cars that are more economical.

Confidence Interval for ISO preference line and Ideal Vector Angle

```
#################################### Confidence interval of ideal vector
angle ###########################
set.seed(123)
final_data <- cbind(y,as.data.frame(z))
nn <- nrow(final_data)
```

```r
### Count of resmapling
bb <- 1000

### Initializing matrices
Z1 <- matrix(0, bb, 1)
Z2 <- matrix(0, bb, 1)
Z3 <- matrix(0, bb, 1)
Z4 <- matrix(0, bb, 1)

### Data Bootstrap 1000 times to get 95% CI for betas
for(ii in 1:bb) {
  data.star <- final_data[sample(nn, nn, replace = T),]
  ### Regression Model - partworths
  out.star <- lm(`Overall Preference` ~ ., data = data.star)
  Z1[ii] <- out.star$coefficients[2]
  Z2[ii] <- out.star$coefficients[3]
  Z3[ii] <- out.star$coefficients[4]
  Z4[ii] <- out.star$coefficients[5]
}

### Slopes of iso-preference and ideal vector at 2.5%
b1 = as.vector(c(sort(Z1)[25]))
b2 = as.vector(c(sort(Z3)[25]))
slope.iso.preference_25 = - b1/b2
slope.ideal.vector_25 = b2/b1

### Angles of iso-preference and ideal vector   at 2.5%
angle.iso.preference_25 = atan(slope.iso.preference_25)*180/pi
angle.iso.preference_25

## [1] 89.75063

angle.ideal.vector_25 = atan(slope.ideal.vector_25)*180/pi
angle.ideal.vector_25

## [1] -0.2493671

### Slopes of iso-preference and ideal vector   at 97.5%
b1 = as.vector(c(sort(Z1)[975]))
b2 = as.vector(c(sort(Z3)[975]))
slope.iso.preference_975 = - b1/b2
slope.ideal.vector_975 = b2/b1

### Angles of iso-preference and ideal vector   at 97.5%
angle.iso.preference_975 = atan(slope.iso.preference_975)*180/pi
angle.iso.preference_975

## [1] -41.98345
```

```r
angle.ideal.vector_975 = atan(slope.ideal.vector_975)*180/pi
angle.ideal.vector_975

## [1] 48.01655

print(paste0("Range of iso preference line angle = [",angle.iso.preference_25
,",",angle.iso.preference_975,"]"))

## [1] "Range of iso preference line angle = [89.7506329087434,-
41.9834492531988]"

print(paste0("Range of ideal vector angle =
[",angle.ideal.vector_25,",",angle.ideal.vector_975,"]"))

## [1] "Range of ideal vector angle = [-0.249367091256649,48.0165507468012]"
```