KBhave_621FinalProject.Rmd

Regression Models : Predicting Glass Type

The CrystalGazer : Kumudini Bhave

May 25, 2017

Contents

1 Regression Models : Glass Type Prediction

1.1 Objective

This is an R Markdown document for providing documentation for performing **Regression** by **Data Exploration, Transformation, Analysis And Modelling and Prediction for the Glass data** to predict the type of glass based on given certain properties of the glass.

This dataset has been taken from the UCI machine learning datasets website https://archive.ics.uci.edu/ml/datasets/glass+identification.

I have chosen this dataset to study the regressions for more than one values of response variable. This data has been recorded in 1987. The primary motivation of the study of different types of glass has been criminal investigation. A curiosity to understand and explore how a study of glass types could lead to insights to satisfy such motivation leads me to choose this for my final project study.

The analysis for this study is organized as follows:

1. Generate several data visualizations to understand the underlying data;
2. Perform data transformations as needed;
3. Develop models to help classify data and Apply models to test data
4. Compare models.

1.2 Background

Glass can be found in most localities. It is produced in a wide variety of forms and compositions, and these affect the properties of this material. It can occur as evidence when it is broken during the commission of a crime. Broken glass fragments ranging in size from large pieces to tiny shards may be transferred to and retained by nearby persons or objects. The mere presence of fragments of glass on the clothing of an alleged burglar in a case involving entry through a broken window may be significant evidence if fragments are found. The significance of such evidence will be enhanced if the fragments are determined to be indistinguishable in all measured properties from the broken window. On the other hand, if the recovered fragments differ in their measured properties from the glass from the broken window, then that window can be eliminated as a possible source of the glass on the subject's clothing.

Glass is technically defined as "The inorganic product of fusion which has cooled to a rigid condition without crystallizing"

Some specialty glasses such as optical glass, novelty glass, or glass that is difficult to melt are produced in pot furnaces or day tanks.

Laminated glass is produced by heat-sealing thin layers of plastic between two or more panes of heat-strengthened glass. In the United States, laminated glass must be installed in the windshields of vehicles, and tempered glass must be installed in the side and rear windows

When a glass object breaks, fragments can be ejected from the object in all directions (Pounds and Smalldon 1978), including backward toward the direction of the breaking force (Nelson and Revell 1967). During experimental studies, glass fragments have been recovered from up to four meters away from a breaking glass object (Francis 1993; Locke and Unikowski 1991). Glass fragments can be transferred onto anything within this distance.

The number of glass fragments that can be transferred is controlled by a number of factors:

-The closer something is to the breaking glass, the more likely it is to have glass fragments transferred to it (Allen and Scranage 1998). The number of fragments transferred decreases with distance from the break (Pounds and Smalldon 1978).

-The person breaking a window will have more glass on him or her than a bystander, and the more blows required to break out the glass, the more glass that will be transferred (Allen et al. 1998b).

-The number of glass fragments generated by a break is independent of the size and thickness of the window but increases with greater damage to the glass (Locke and Unikowski 1992).

-Whether the glass that is deposited on clothing persists to be recovered by a forensic examiner depends upon additional factors:

-Less glass is retained on slick clothing, such as nylon jackets, than on rough clothing, such as wool sweaters. Wet clothing retains more glass than dry clothing (Allen et al. 1998b).

-Glass fragments fall off clothing over time, and larger pieces fall off before smaller pieces (Cox et al. 1996a, 1996b, 1996c; Hicks et al. 1996; Hoefler et al. 1994).

-Glass falls off faster if the person wearing the clothing is active (Batten 1989; Cox et al. 1996c; Hicks et al. 1996). It should be noted, however, that the transfer and persistence of glass is highly variable

A primary transfer-is a transfer from the broken glass object to something else. Primary transfer also can occur when a person or object comes into contact with previously broken glass (Allen et al. 1998a). Additionally, there can be secondary transfer of glass between people and objects, such as when glass is transferred from a person to a vehicle sea During a glass examination, it cannot be positively determined whether the glass fragments found on an object were acquired through primary transfer, secondary transfer, or through contact with previously broken glass

Optical Properties such as Refractive index (n):

It is a unitless measure of the speed of light in a transparent medium and is defined by Snell's law as the ratio of the velocity of light in a vacuum to the velocity of the wave in the transparent medium (Stoiber and Morse 1981). Refractive index is a function of chemical composition and atomic arrangement Refractive index is the most commonly measured property in the forensic examination of glass fragments (Koons et al. 2002), because: - Precise refractive indices can be measured rapidly on the small fragments typically found in casework. - It can aid in the characterization of glass. - It provides good discrimination potential. (Koons et al. 2002)

Elemental Analysis such as chemical composition of glass :

Manufacturers control the concentrations of many chemical elements to impart specific properties to their glass product. The chemical analysis remains the best means for differentiating glass specimens

The Crime Investigation Aspect :

Glass is formidable, forensically speaking. Formidable because - despite the fact that once smashed, it fragments into millions of tiny shards, from the large to the microscopic - it is nearly impossible for any suspect involved in a scene where it is present not to take away a trace. A glass impact will shower fragments into a surprisingly large radius - up to around three metres. If a forensic investigator trained in glass recovery manages to trace an intruder and seize some clothes - if he was wearing them at the time - there's a significant chance it will have glass that would tie him to the scene in question. Glass can also be found in some unlikely places too - like our burglar's hair, ears, underneath the fingernails or even in the skin.

1. Looking back: the refractive index

This highly accurate test relies on the reflectivity of the recovered and control samples being compared. Light is shone on the recovered glass and a control sample from the window itself. The test measures how much light is reflected and how much is absorbed by the sample. If it matches that of the control sample - its likely to be from the same window or fixture.

2. The stuff inside: chemical analysis

Analysing the specimen sample against the control using scanning electron microscopy enables an astute investigator to determine if the physical constituents of the samples are the same. It provides detailed information on the elements used in the glass's composition, such as the composition of sand and other chemicals. If the physical composition of the specimen matches the control sample - it is likely to be from the same place.

Glass is also immensely useful, forensically speaking, in other ways. If a bullet was fired through a window, for example, the pattern of glass fracture may tell us the trajectory from which it entered. A burglar may cut himself on a broken window entering or exiting a building, leaving tell-tale DNA evidence at the scene. And as every frustrated window cleaner knows, glass is great for showing up everyone's favourite piece of tell tale evidence - fingerprints.

Glass analysis, hence, is another invaluable tool in our arsenal for linking suspects to a crime scene.

1.3 Technical Summary

This analysis demonstrates several analytic techniques to examine various the optical and chemical properties of the glass, and classify the glass type. The various techniques used in this study are multinomial logistic regression, random forests, conditional inference trees, linear discriminant analysis. The best performing model , Random Forest model, gave an accuracy of 86%.

The most influential predictos :????

To facilitate the model selection through crossvalidation we will be randomly dividing the glass data into a training set that contains 70% of the data and a test set that contains 30% of the data.

1.4 Exploratory Data Analysis

The first step in any analysis is to obtain the dataset and codebook. Both the dataset and the codebook can be downloaded for free from the UCI website.

1.4.1 Glass DataSet Evaluation

The Glass data set contains 214 cases, including an indentification variable (INDEX), 9 predictors, and one response variable. Each case is a commercially available glass, with the response variable being the type of glass, depending upon the chemical composition and the refractive index.

Of the 9 predictor variables, 8 are related to chemical properties ofthe glass and 1 is the physical property of refractive index of glass.

The response variable is:

GlassType: This indicates the type of glass.

The various potential predictor variables in the original dataset under study are :

Predictor Variables	Definition
Id	number: 1 to 214
RI	Refractive Index
Na	Sodium (unit measurement: weight percent in corresponding oxide,
Mg	Magnesium
Al	Aluminum
Si	Silicon
K	Potassium
Ca	Calcium
Ba	Barium
Fe	Iron
Type of glass	(class attribute) Values given table below

Value	Type Of Glass
1	building_windows_float_processed
2	building_windows_non_float_processed
3	vehicle_windows_float_processed
4	vehicle_windows_non_float_processed (none in this database) 5
5	containers
6	tableware
7	headlamps

1.4.2 Loading the Dataset :

```
knitr::opts_chunk$set(message = FALSE, echo = TRUE)


# Library for data display in tabular format

# library(DT)
library(tidyr)
library(dplyr)
library(readr)
# For string extractions
library(stringr)
# Library for plotting
library(ggplot2)
library(gridExtra)
library(Amelia)
library(ggmosaic)

library(corrplot)
library(e1071)
library(data.table)
# statistical packages
library(knitr)
library(caret)
library(pander)
# ROC
library(glmnet)
library(mlogit)
library(pROC)
library(car)
library(bestglm)
library(vcd)
library(MASS)
library(forecast)
# Loading RCurl package to help scrape data from web (stored on GitHub).
library(RCurl)

# Loading plyr package to help map abbreviated values to explained.
library(plyr)
```

```
# Getting data

### Extracting Raw Data From GitHub Data File And Reading In CSV Format

data.giturl <- "https://raw.githubusercontent.com/DataDriven-MSDA/DATA621/master/Final/glass.data"
glass.gitdata <- getURL(data.giturl)
glass.gitdata.csv <- read.csv(text = glass.gitdata, header = F, sep = ",", stringsAsFactors = FALSE)

# View(glass.gitdata.csv)
```

1.4.3 Renaming Attributes

We name the attributes per the data description/codebook.

```
# Naming the Attributes selected for study
colnames(glass.gitdata.csv) <- c("ID", "RI", "Na", "Mg", "Al", "Si", "K", "Ca", "Ba",
    "Fe", "GlassType")



# Verifying the number of attributes
length(glass.gitdata.csv)
# Verifying the number of observations selected
nrow(glass.gitdata.csv)
```

I have removed the Id column and work only with potential predictors.

```
traindataorig <- dplyr::select(glass.gitdata.csv, -1)


nrow(traindataorig)
ncol(traindataorig)
# View(traindataorig) View(traindata) View(evaldata) head(traindataorig)
```

1.4.4 Data Exploration

Below is the summary of the potential predictor variables and the response variable "GlassType" in the dataset.

1.4.4.1 Response Variable:

GlassType

It is found that the "GlassType" response variable has discrete values ranging from 1 to 7.Each Type represent whether it is float processed or not, and if it for vehicle or window , container, tableware or lamp. The variable is categorical in nature. There is no particular order for these values. GlassType has 214 observations of glass which constitutes float processed glass for windows constitutes 32.71% of the total observations. The highest number of glass type is Type 2 which is non float processed windows glass constituting about 35.51%

Least number is that of Type 6 which is 4.2% for tableware type of glass.

```
pander(table(traindataorig$GlassType), caption = "Response Variable : GlassType")
```

Table 3: Response Variable : GlassType

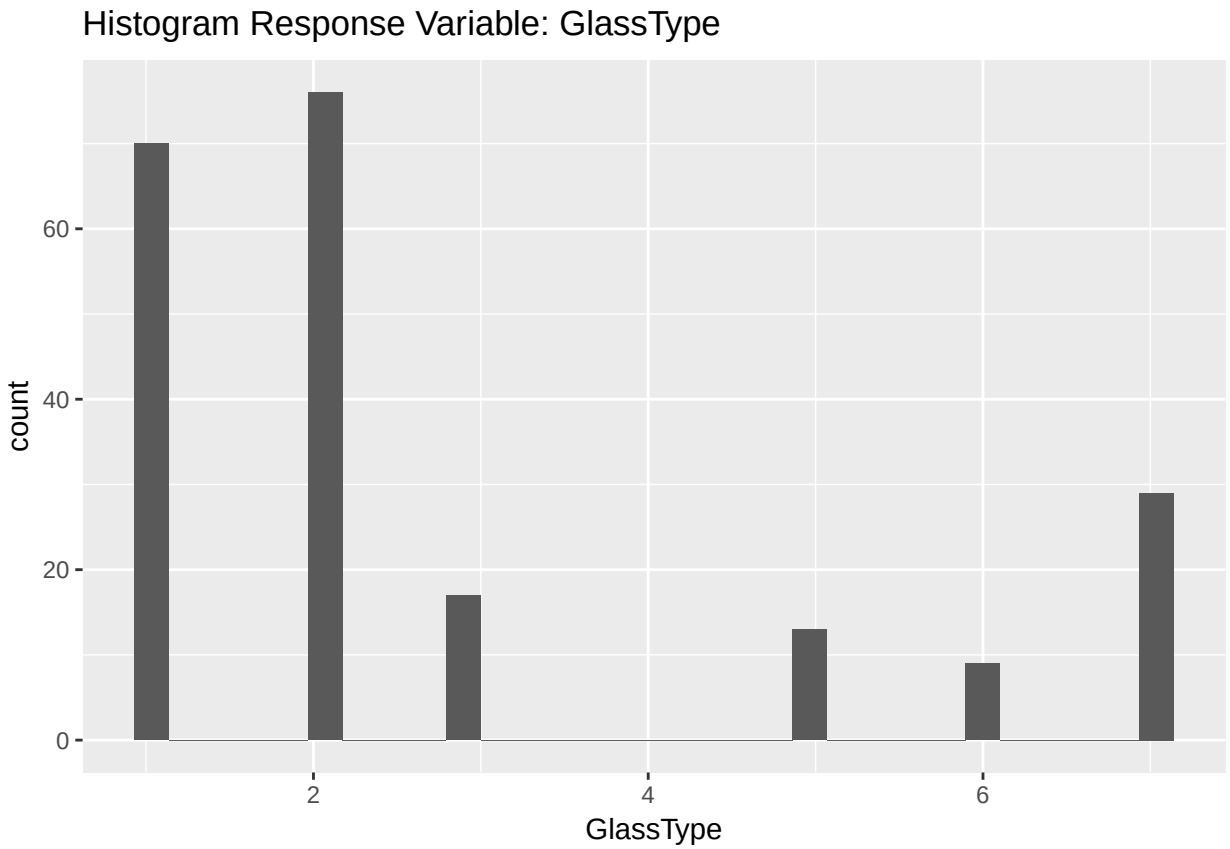
1	2	3	5	6	7
70	76	17	13	9	29

```
pander(table(traindataorig$GlassType)/sum(table(traindataorig$GlassType)), caption = "Frequency Table :
```

Table 4: Frequency Table : GlassType

1	2	3	5	6	7
0.3271	0.3551	0.07944	0.06075	0.04206	0.1355

```
# Histogram for the response variable GlassType par(mfrow=c(1,2))
ggplot(traindataorig, aes(x = GlassType)) + geom_histogram() + ggtitle("Histogram Response Variable: Gl
```



1.4.4.2 Predictor Variables :

Let's take a look at the summary of the potential predictor variables.

```
# Summary Table

means <- sapply(traindataorig, mean)
medians <- sapply(traindataorig, median)
IQRs <- sapply(traindataorig, IQR)
skews <- sapply(traindataorig, skewness)
sds <- sapply(traindataorig, sd)
cors <- as.vector(cor(traindataorig$GlassType, traindataorig[, 1:ncol(traindataorig)]))

header <- (c("MEAN", "MEDIAN", "IQR", "SKEWNESS", "STD.DEV", "CORRELATION"))

datasummary <- as.data.frame(cbind(means, medians, IQRs, skews, sds, cors))
datasummary <- round(datasummary, 2)
colnames(datasummary) <- header
pander(datasummary, caption = "Summary of Glass Data Variables")
```

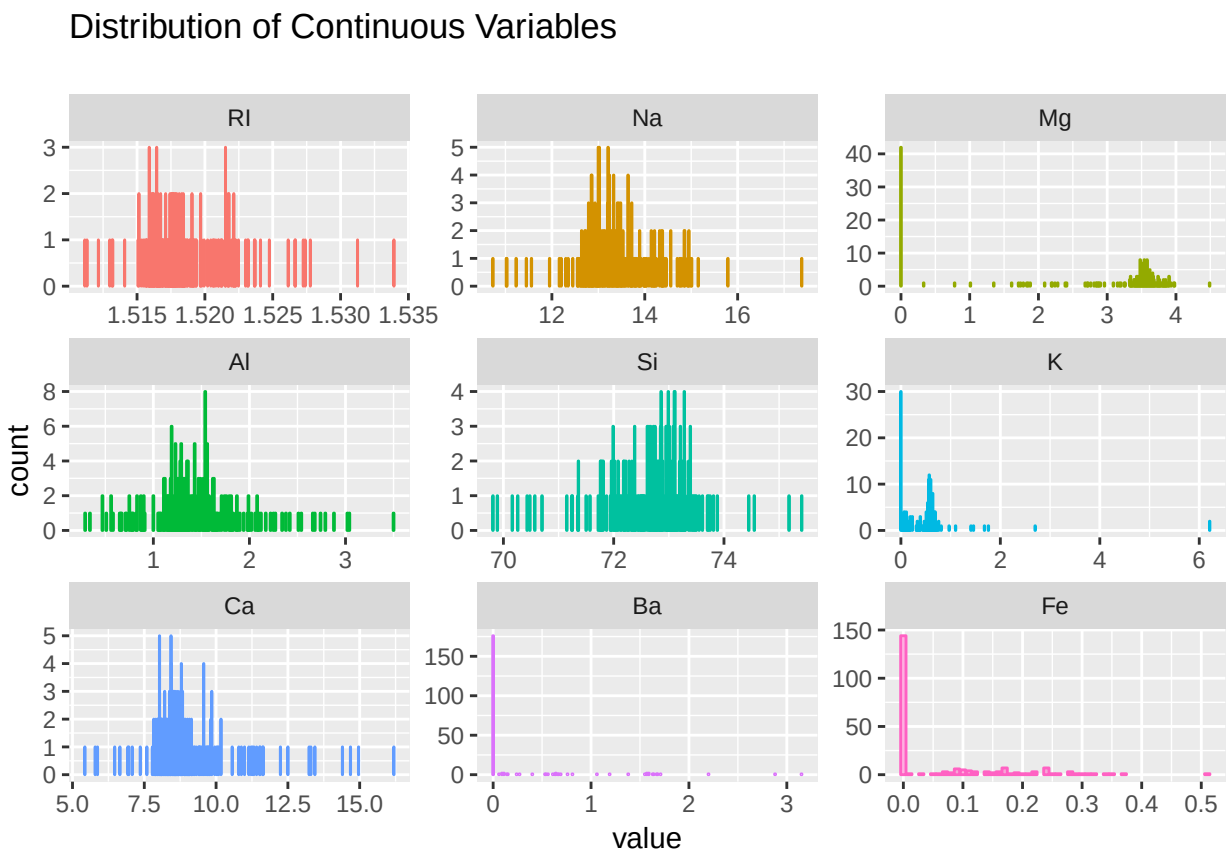
Table 5: Summary of Glass Data Variables

	MEAN	MEDIAN	IQR	SKEWNESS	STD.DEV	CORRELATION
RI	1.52	1.52	0	1.6	0	-0.16
Na	13.41	13.3	0.92	0.45	0.82	0.5
Mg	2.68	3.48	1.49	-1.14	1.44	-0.74

	MEAN	MEDIAN	IQR	SKEWNESS	STD.DEV	CORRELATION
Al	1.44	1.36	0.44	0.89	0.5	0.6
Si	72.65	72.79	0.81	-0.72	0.77	0.15
K	0.5	0.56	0.49	6.46	0.65	-0.01
Ca	8.96	8.6	0.93	2.02	1.42	0
Ba	0.18	0	0	3.37	0.5	0.58
Fe	0.06	0	0.1	1.73	0.1	-0.19
GlassType	2.78	2	2	1.1	2.1	1

The predictor variables are numerical. Plotted below are the distributions of each of the variables.

```
# Continuous Predictor Distributions

contipred <- traindataorig %>% dplyr::select(-c(GlassType))

meltedc <- melt(contipred)

ggplot(meltedc, aes(value)) + geom_bar(aes(fill = variable, col = variable), alpha = 0.5,
    show.legend = FALSE) + facet_wrap(~variable, scale = "free") + ggtitle("Distribution of Continuous
```



It is observed that the **K** i.e. Potassium, **Ba** i.e Barium, **Fe** i.e. Iron elements are heavily rightskewed. Whether the tranformations of these could help form a better correlation with the response variable GlassType would be further explored. **Ca** Calcium, is very lightly right skewed. The other predictors are roughly normally distributed.

1.4.5 Box Plot for Predictors by GlassType

The distribution of Predictor Variables and the magnitude of the constituency of each chemical component /refractive index in Glass per GlassType can be observed below from the boxplots.

Table 1. Index of Refraction Ranges for Several Types of Glasses

Glass	Index of Refraction
Headlight glass	1.47-1.49
Television glass	1.49-1.51
Window glass	1.51-1.52
Bottles	1.51-1.52
Ophthalmic lenses	1.52-1.53

Table taken from Saferstein, R., Criminalistics Lab Manual, p. 30 (reference above).

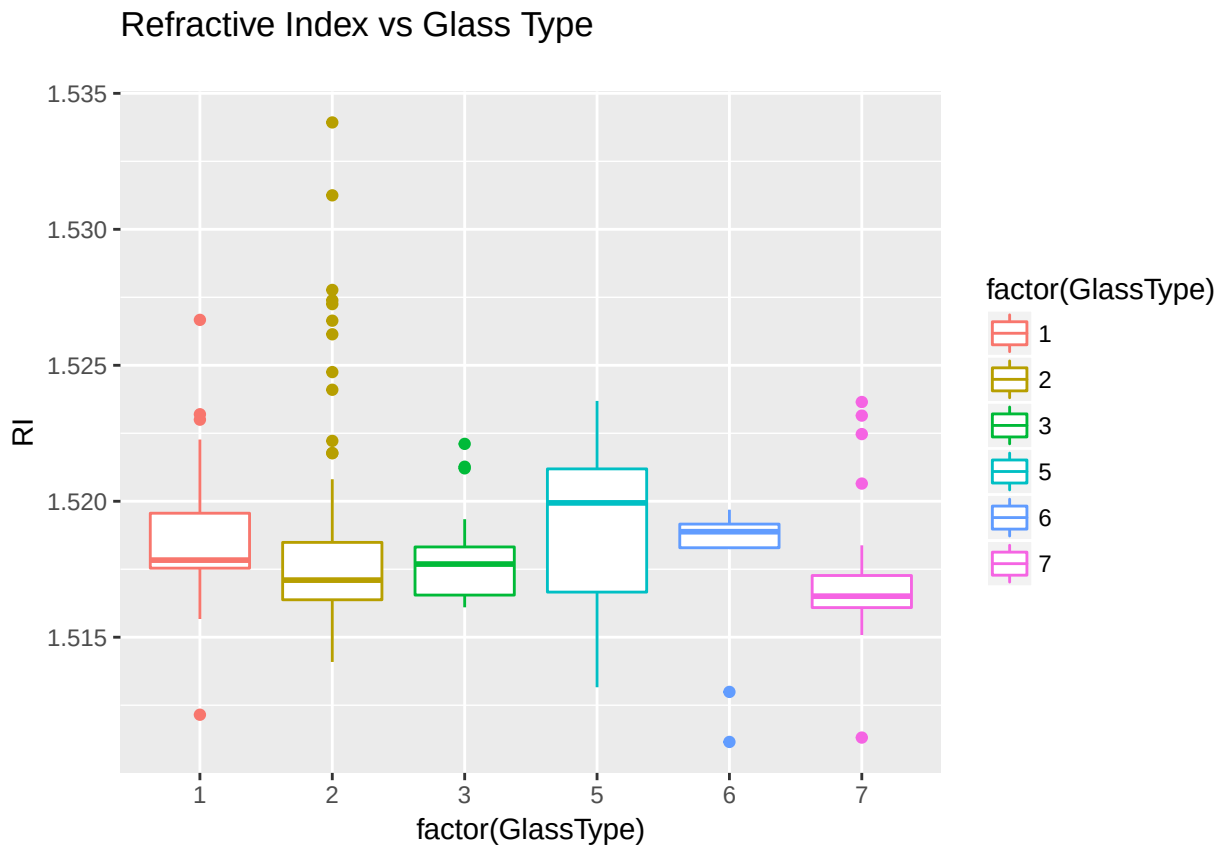
Table 2. Index of Refraction for Specific Glasses

Glass	Index of Refraction
Windshield glass 1	1.518
Windshield glass 2	1.520
Headlight glass from Toyota Celica	1.478
Headlight glass from Chevrolet Eurosport	1.488
Headlight glass from Oldsmobile Cutlass Ciera	1.488
Headlight glass from Toyota Corolla	1.478
Bottle glass from Lipton Iced Tea	1.524

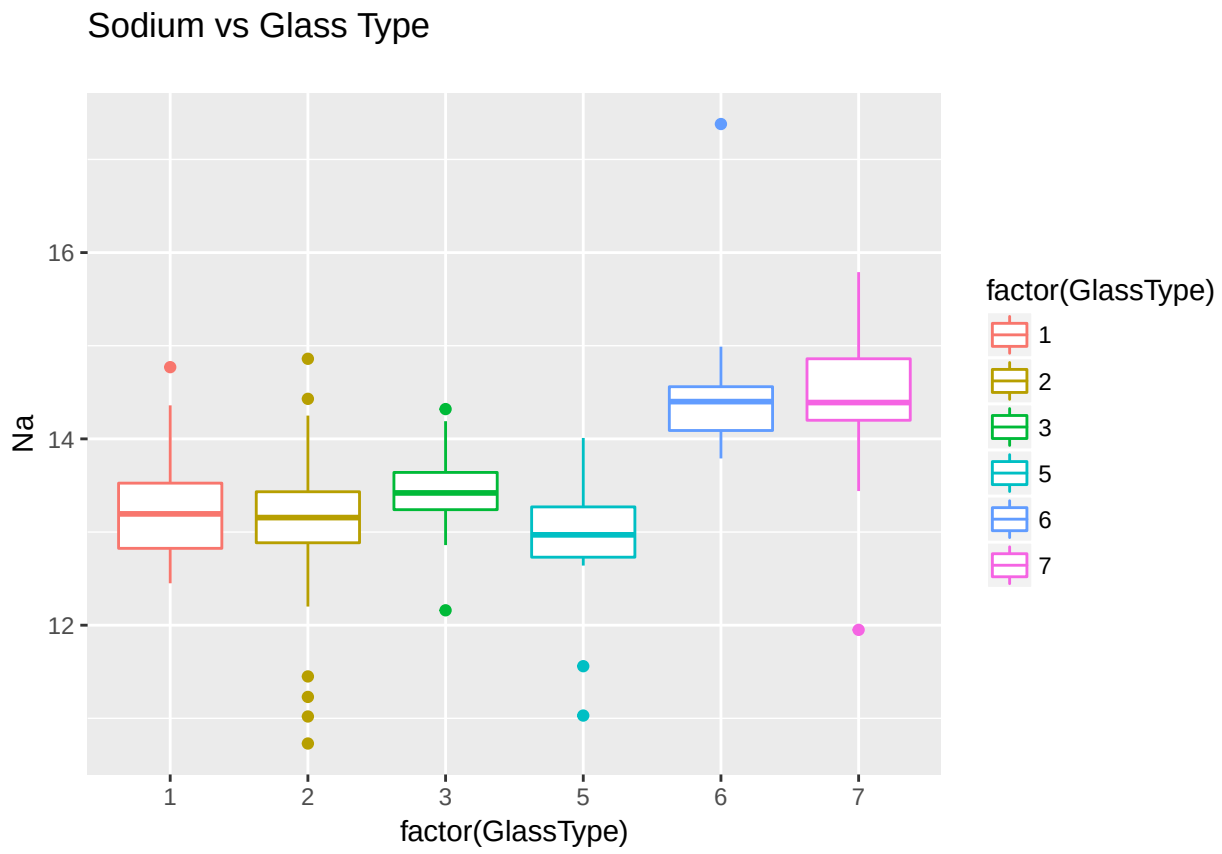
While this database does not contain any data on GlassType 4 which is non float processed vehicle glass, the boxplots show distributions per other type of glasses.

```
# traindataorig$GlassType <- as.factor(traindataorig$GlassType)


RIbox <- ggplot(traindataorig, aes(factor(GlassType), RI, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Refractive Index vs Glass Type\n")

Nabox <- ggplot(traindataorig, aes(factor(GlassType), Na, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Sodium vs Glass Type\n")

Mgbox <- ggplot(traindataorig, aes(factor(GlassType), Mg, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Magnesium vs Glass Type\n")


Albox <- ggplot(traindataorig, aes(factor(GlassType), Al, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Aluminium vs Glass Type\n")

Sibox <- ggplot(traindataorig, aes(factor(GlassType), Si, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Silicon vs Glass Type\n")

Kbox <- ggplot(traindataorig, aes(factor(GlassType), K, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Potassium vs Glass Type\n")
```

```
Cabox <- ggplot(traindataorig, aes(factor(GlassType), Ca, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Calcium vs Glass Type\n")

Babox <- ggplot(traindataorig, aes(factor(GlassType), Ba, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Barium vs Glass Type\n")

Febox <- ggplot(traindataorig, aes(factor(GlassType), Fe, colour = factor(GlassType))) +
    geom_boxplot() + ggtitle("Iron vs Glass Type\n")

par(mfrow = c(1, 2))
RIbox
```



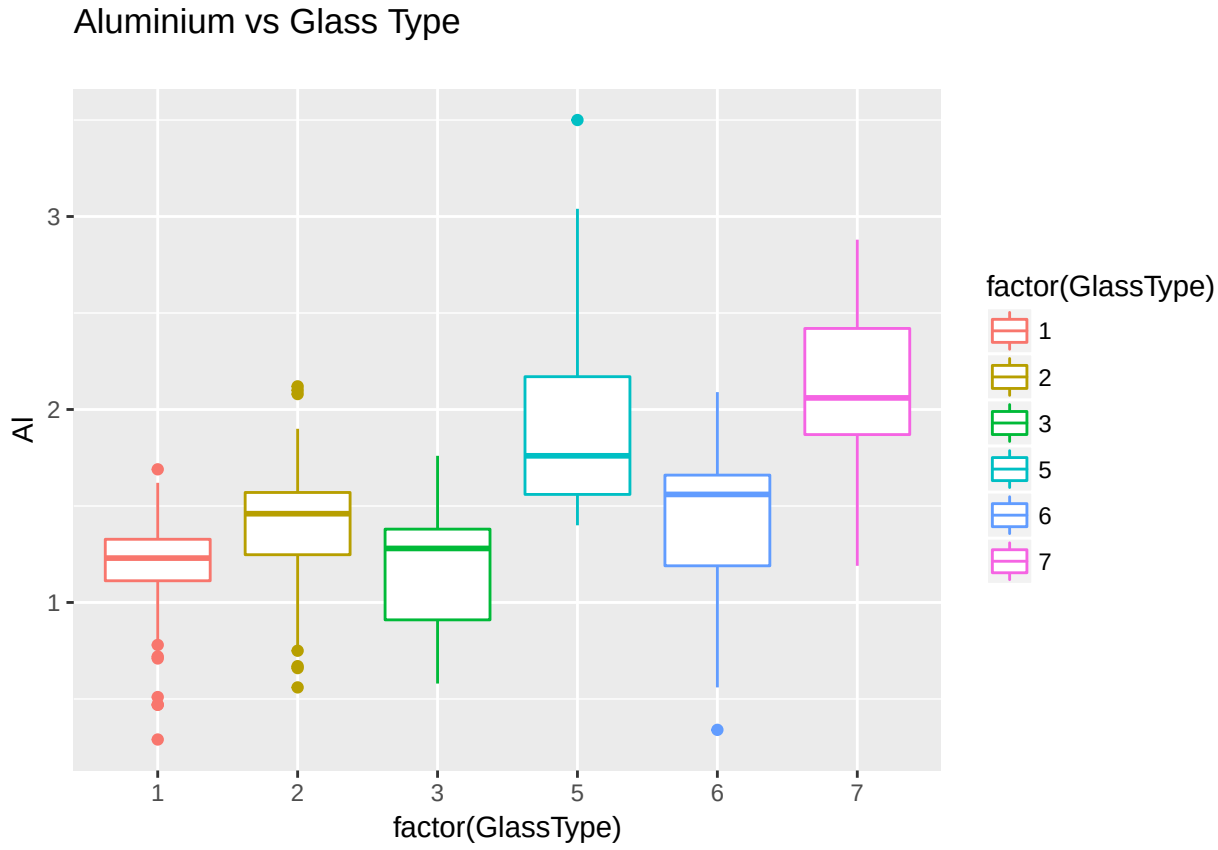

```
Nabox
```



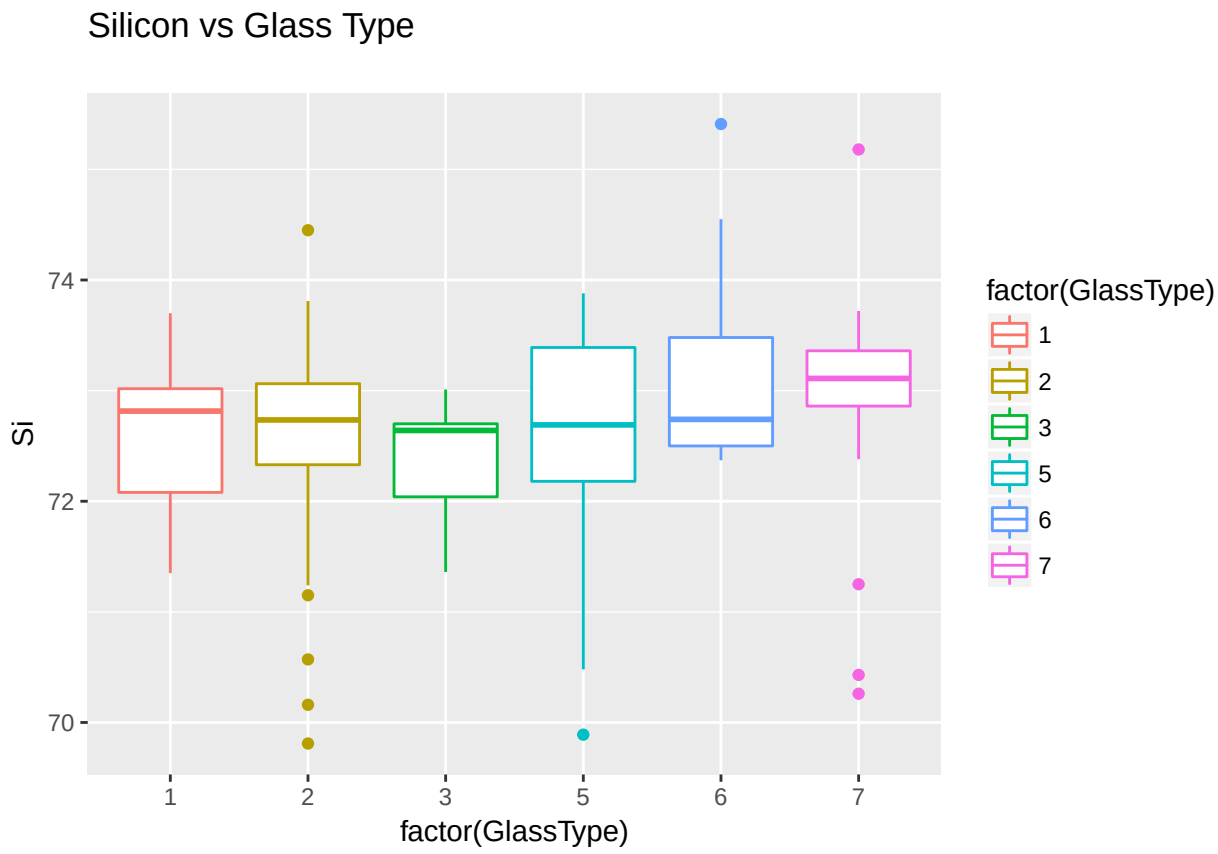

```
par(mfrow = c(1, 2))
Mgbox
```


Magnesium vs Glass Type
Mg
4
3
2
1
0
1
2
3
5
6
7
factor(GlassType)
factor(GlassType)
1
2
3
5
6
7

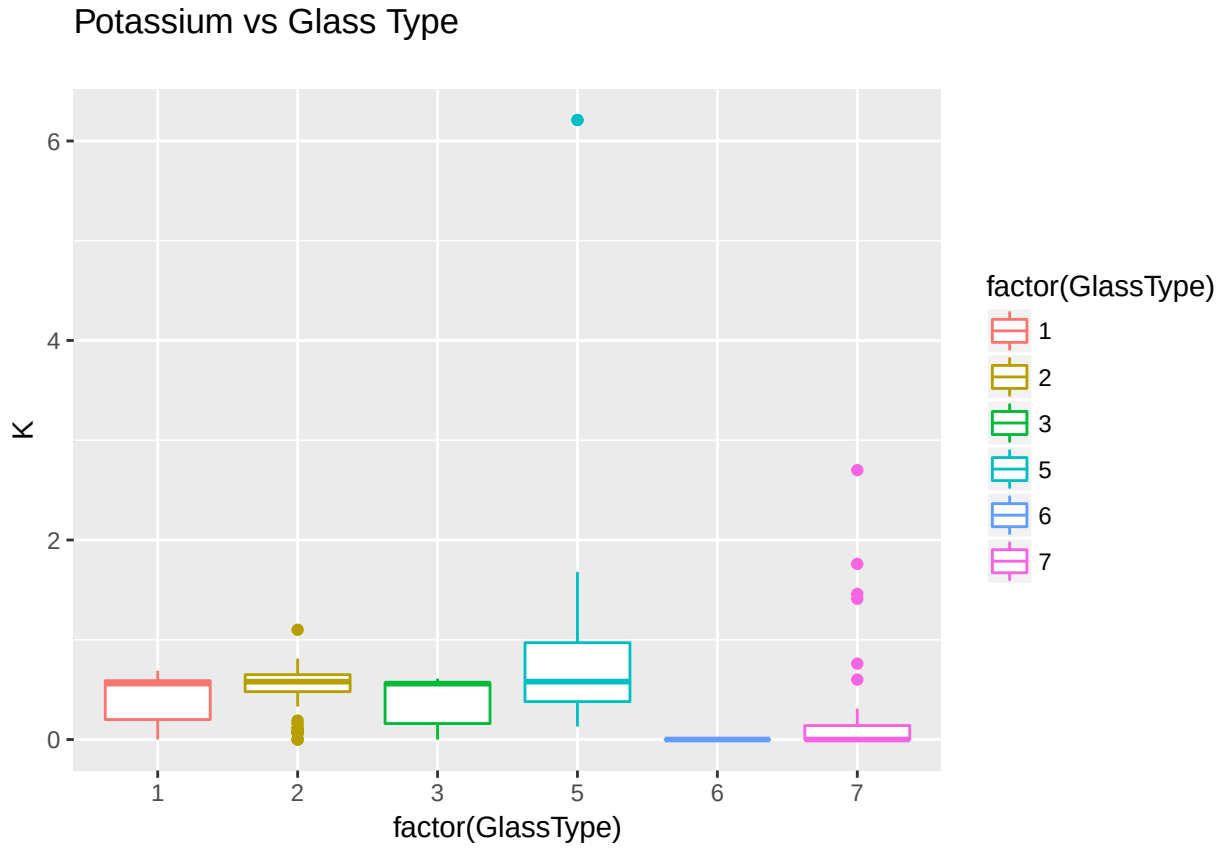
```
Albox
```



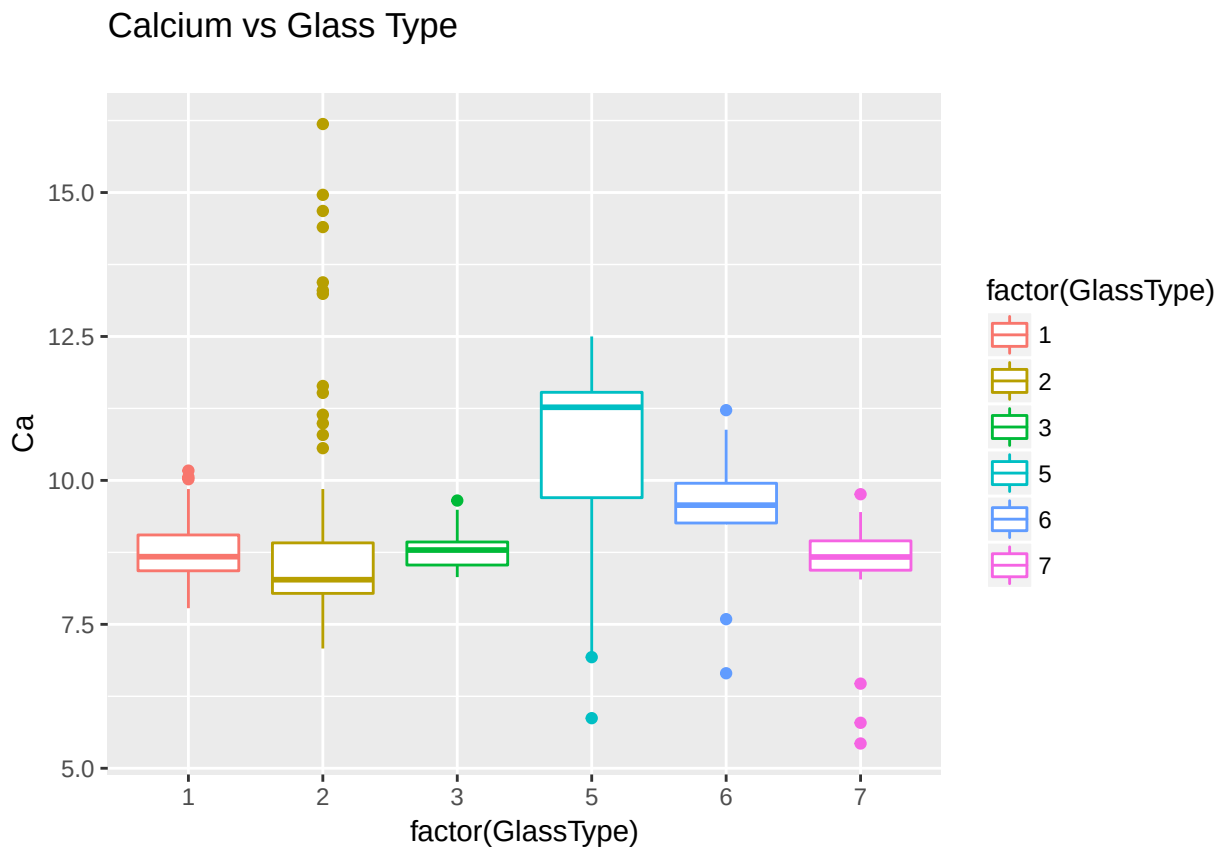

```
par(mfrow = c(1, 2))
Sibox
```


Silicon vs Glass Type
Si
74
72
70
1
2
3
5
6
7
factor(GlassType)
factor(GlassType)
1
2
3
5
6
7

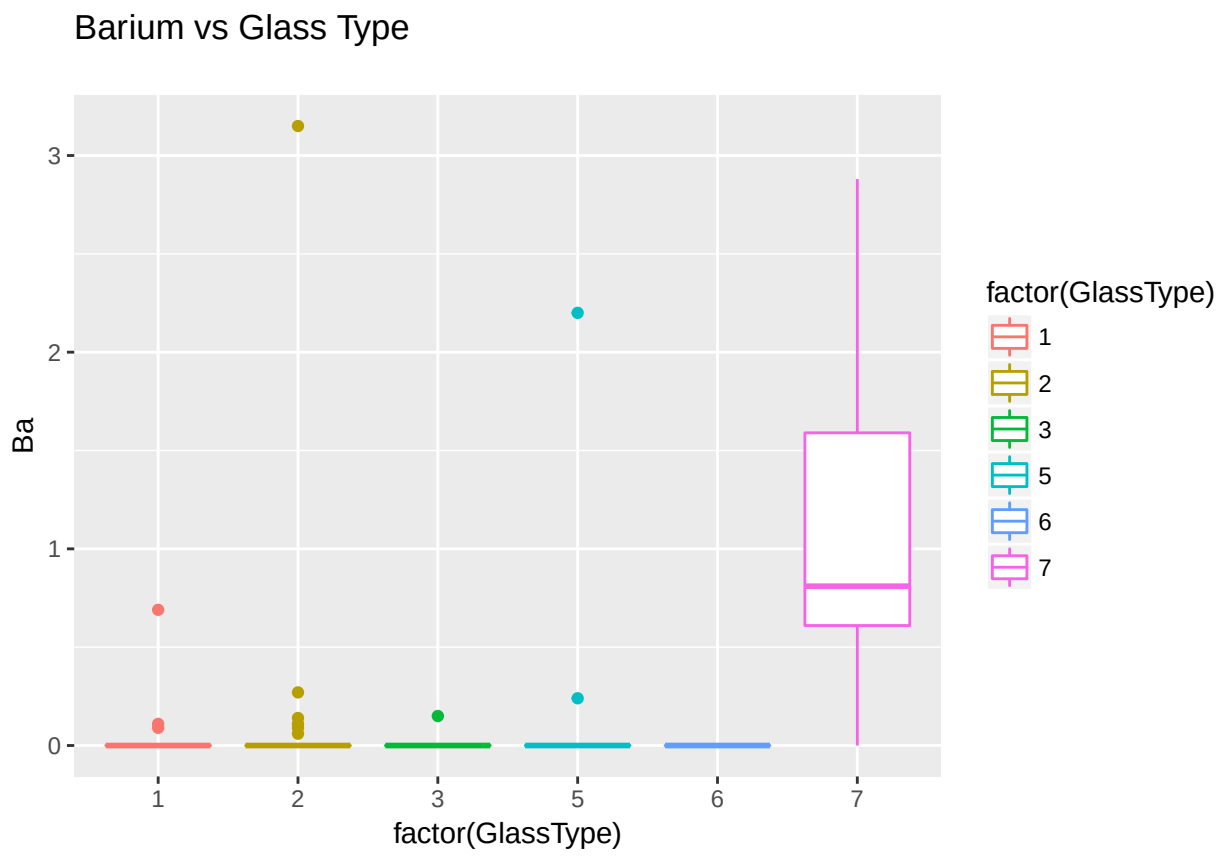
```
Kbox
```



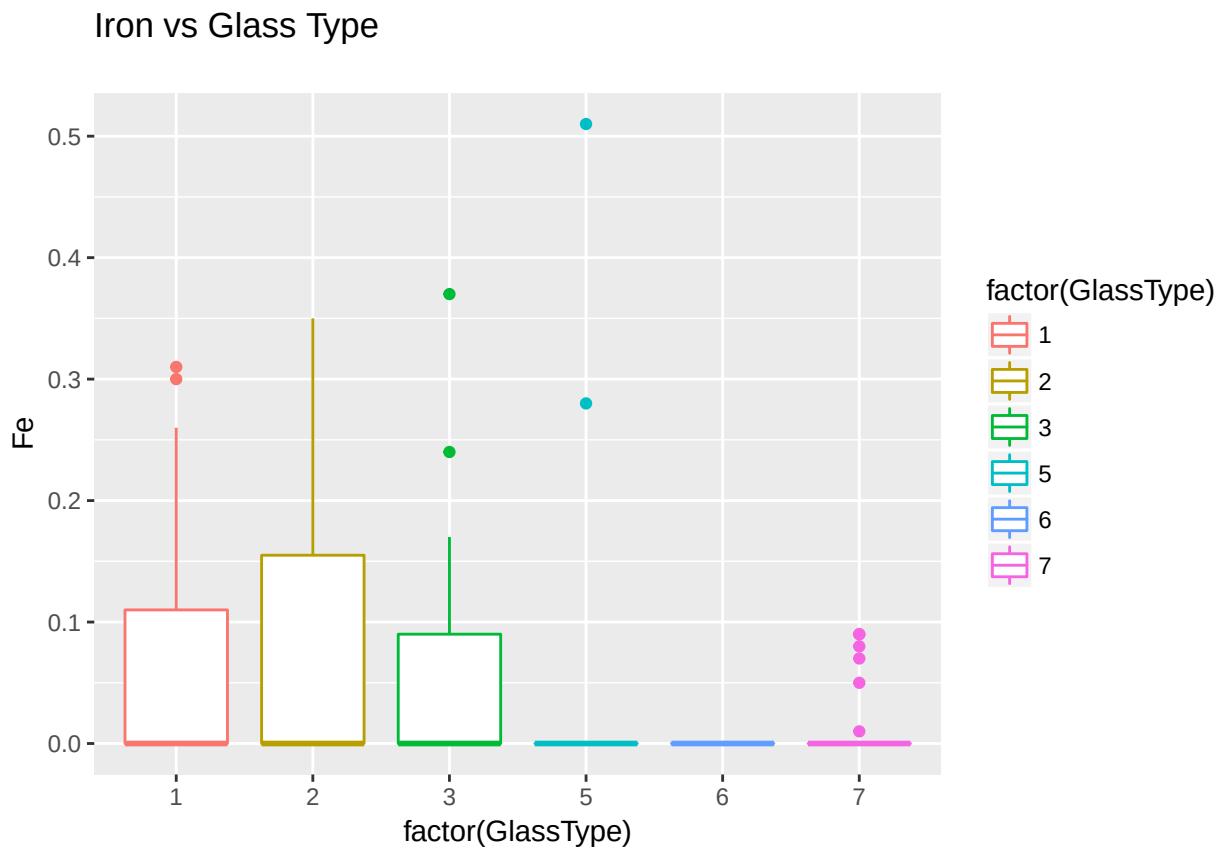

```
par(mfrow = c(1, 2))
Cabox
```


Calcium vs Glass Type
15.0
12.5
10.0
7.5
5.0
Ca
1
2
3
5
6
7
factor(GlassType)
factor(GlassType)
1
2
3
5
6
7

```
Babox
```


Barium vs Glass Type
Ba
0
1
2
3
1
2
3
5
6
7
factor(GlassType)
factor(GlassType)
1
2
3
5
6
7

```
Febox
```




```
# grid.arrange(RIbox,Nabox,Mgbox,Albox,Sibox,Kbox,Cabox,Babox,Febox,ncol=3,nrow=3)
```

From the Table above , it is intuitive that the higher **refractive index** is associated with container type of glass as compared to other types of glasses and that is what is observed in boxplot of RI vs GlassType for Type 5 glass.

It is intuitive that **Sodium (Na)** content is least for the container type of glasses (as observed in box plot of Na) as Container glass is a soda-lime glass that is a slight variation on flat glass, which uses more alumina and calcium, and less sodium and magnesium which are more water-soluble. This makes it less susceptible to water erosion. It is high for the Type 6 tableware and Type 7 headlamps

The Vehicle and the Window glasses both float and non float processed are high in Magnesium content.

The **Si (Silica) and Al(Aluminium)** in its Alumina form stand heat expansion much better than window glass.[10] Used for chemical glassware, cooking glass, car head lamps, etc. Borosilicate glasses (e.g. Pyrex, Duran) have as main constituents silica and boron trioxide. They have fairly low coefficients of thermal expansion. Expectedly these seem to be higher for container , tableware applications.

Stabilizers make the glass strong and water resistant. Calcium carbonate, often called calcined limestone, is a stabilizer. Without a stabilizer, water and humidity attack and dissolve glass. It is observed that **Ca** Calcium is higher in container type of glass.

The Ba Barium content of glass is for the crystalline nature and is an alternative material to lead which can have health hazards. Alkali-barium Silicate Glass are also use in Televisiosn. Without this type of glass, watching TV would be very dangerous. A television produces X-rays that must be absorbed, otherwise they could in the long run cause health problems. The X-rays are absorbed by glass with minimum amounts of heavy oxides (lead, barium or strontium). Lead glass is commonly used for the funnel and neck of the TV tube, while glass containing barium is used for the screen. From the boxplots, it is observed to have high potential in headlamp glass.

1.5 Data Preparation

1.5.1 Missing Data

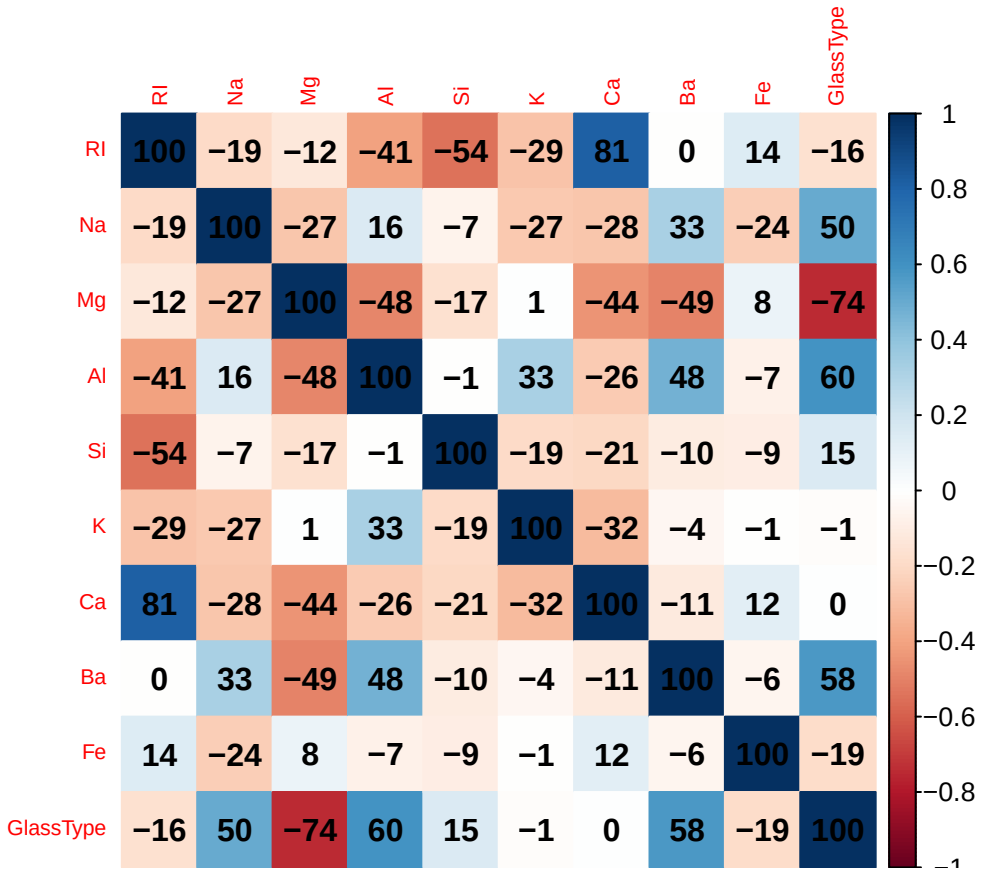
The data is pretty clean with no NAs and no outliers. But this dataset does not have any data for Type 4 Vehicle non float processed glass.

1.5.2 Correlation Matrix.

The correlation matrix shows **Na** Sodium , **Mg** Magnesium, **Al** Aluminium , **Ba** are the most correlated in deciding the GlassType. The **K** Potassium content , **Ca** calcium content are not so significantly coorelated. **Si** Silica, **Fe** Iron, and physcial Refractive Index property have some amount of correlation to the GlassType.

In the models built , I will see how much of these physical/chemical properties hold true significance in determining and correctly predicting the GlassType.

```
cormat <- as.matrix(cor(traindataorig, use = "pairwise.complete.obs"))
corrplot(cormat, method = "color", tl.cex = 0.7, addCoef.col = "black", addCoefasPercent = TRUE)
```



1.5.3 Transformations

Here, I have tried to see the BoxCox method suggests any transformations. Below is a table depicting the possible log, sqrt, reiprocal transformations for the variables and possible change/no change in the correlation factor w.r.t the response variable GlassType.

```
# BOXCOX lambda computing for each variable

lambda.RI <- BoxCox.lambda(traindataorig$RI)  # Indicates Using As Is , Y^1

lambda.Na <- BoxCox.lambda(traindataorig$Na)  # Indicates Using As Is , Y^1

lambda.Mg <- BoxCox.lambda(traindataorig$Mg)  # Indicates Using As Is , Y^1

lambda.Al <- BoxCox.lambda(traindataorig$Al)  # Indicates Using As Is , Y^1

lambda.Si <- BoxCox.lambda(traindataorig$Si)  # Indicates Using As Is , Y^1

lambda.K <- BoxCox.lambda(traindataorig$K)  # Indicates Using As Is , Y^1

lambda.Ca <- BoxCox.lambda(traindataorig$Ca)  # Indicates Using As Is , Y^1

lambda.Ba <- BoxCox.lambda(traindataorig$Ba)  # Indicates Using As Is , Y^1

lambda.Fe <- BoxCox.lambda(traindataorig$Fe)  # Indicates Using As Is , Y^1

lambda.GlassType <- BoxCox.lambda(traindataorig$GlassType)  # Indicates Using As Is , Y^1


lambdaX <- rbind(lambda.RI, lambda.Na, lambda.Mg, lambda.Al, lambda.Si, lambda.K,
    lambda.Ca, lambda.Ba, lambda.Fe, lambda.GlassType)
```

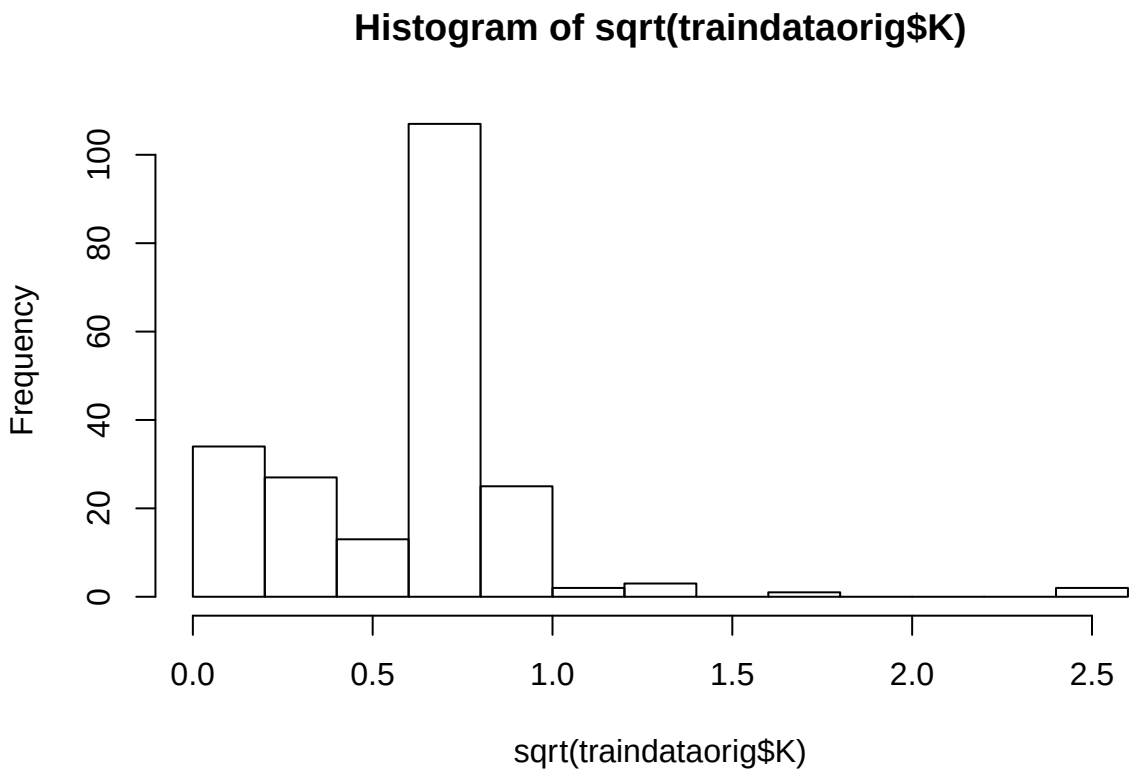
```
# Transformation Matrix, to see the possibility of any improvement in correlation
# to GlassType

cors2 <- as.vector(cor(traindataorig$GlassType, traindataorig[, 1:ncol(traindataorig)]))

log_cors <- as.vector(cor(traindataorig$GlassType, log(traindataorig[, 1:ncol(traindataorig)])))

sqrt_cors <- as.vector(cor(traindataorig$GlassType, sqrt(traindataorig[, 1:ncol(traindataorig)])))

recip_cors <- as.vector(cor(traindataorig$GlassType, (1/traindataorig[, 1:ncol(traindataorig)])))

header2 <- (c("No Transform", "LOG", "SQ.ROOT", "1/X", "Lambda"))

transforms <- as.data.frame(cbind(cors2, log_cors, sqrt_cors, recip_cors, lambdaX))
transforms <- round(transforms, 2)
rownames(transforms) <- colnames(traindataorig)
colnames(transforms) <- header2
pander(transforms)
```

	No Transform	LOG	SQ.ROOT	1/X	Lambda
RI	-0.16	-0.16	-0.16	0.16	-1
Na	0.5	0.48	0.49	-0.46	-0.74
Mg	-0.74	NA	-0.72	NA	1
Al	0.6	0.49	0.56	-0.32	0.36
Si	0.15	0.15	0.15	-0.15	-1
K	-0.01	NA	-0.31	NA	0.06
Ca	0	-0.02	-0.01	0.04	-0.39

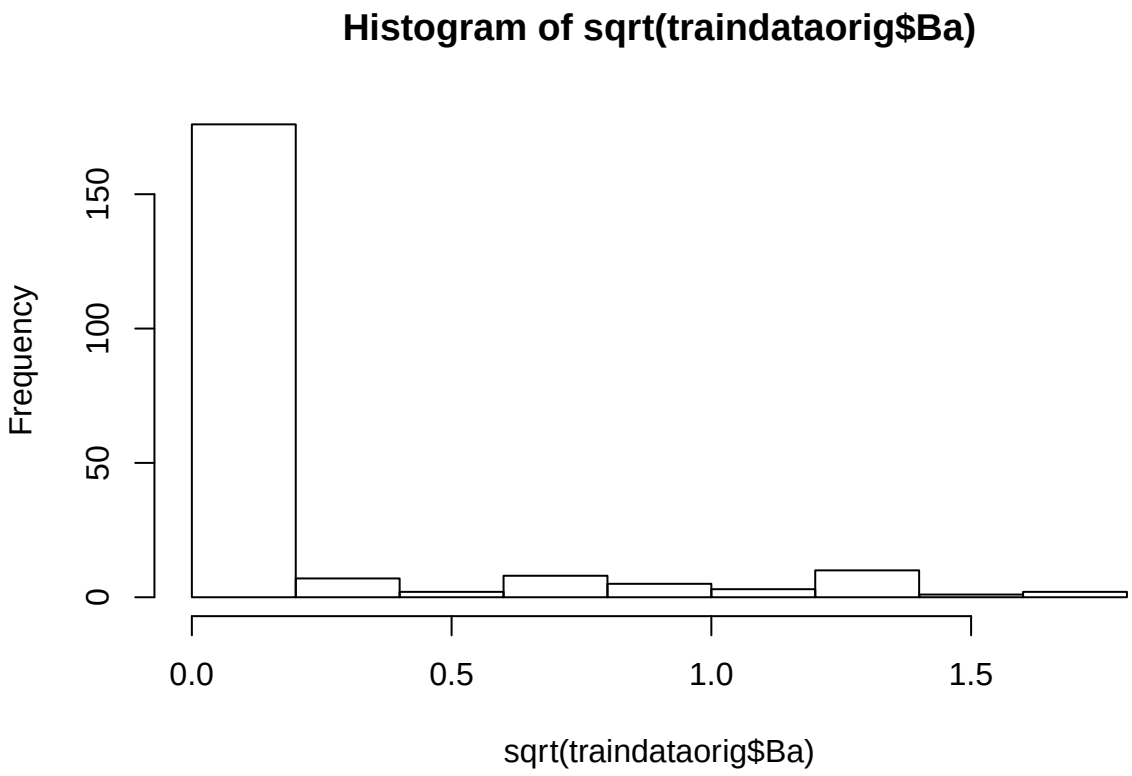
	No Transform	LOG	SQ.ROOT	1/X	Lambda
Ba	0.58	NA	0.66	NA	0.09
Fe	-0.19	NA	-0.21	NA	0.13
GlassType	1	0.96	0.99	-0.85	2

For the skewed distributions of **K** Potassium, **Ba** Barium, **Fe** Iron, it is observed that there is some improvement with square root transformation of **K**, **Ba** and **Fe** variables. I will go ahead and do these to explore them in models.

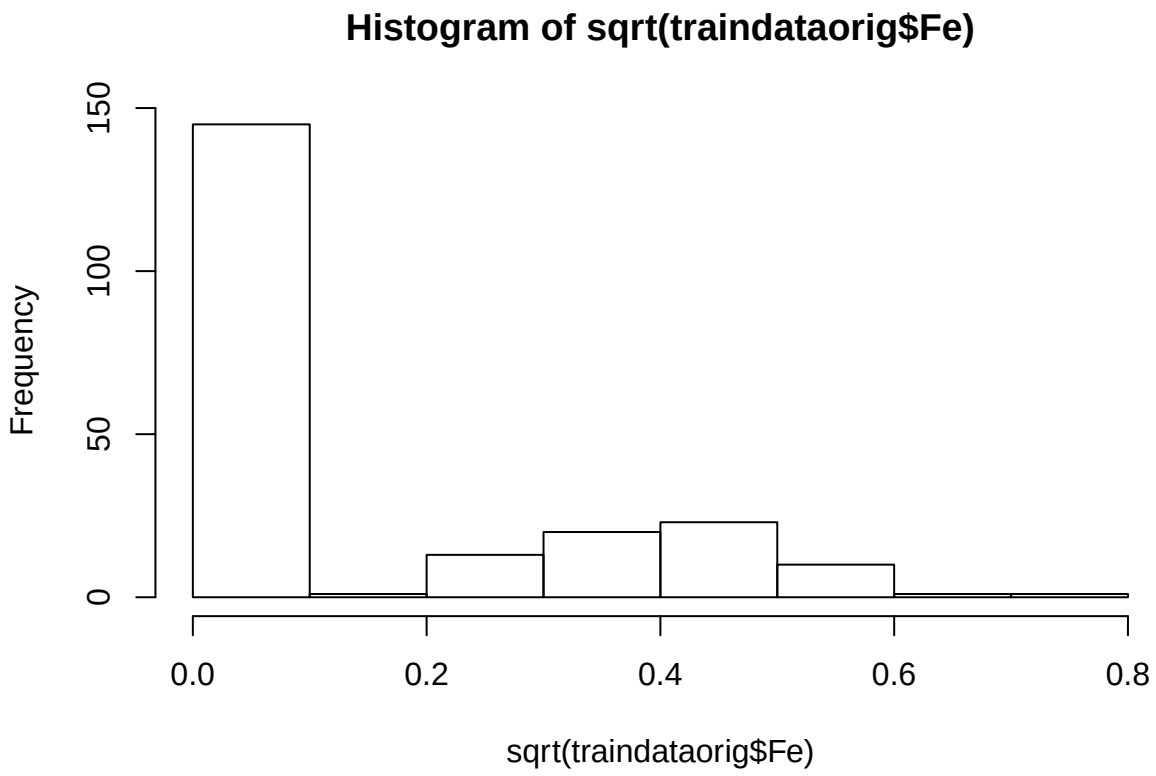
```
# Transformation applied to these three predictor variables

traindataorig$K.sqrt <- sqrt(traindataorig$K)
hist(sqrt(traindataorig$K))  # select this
```




```
traindataorig$Ba.sqrt <- sqrt(traindataorig$Ba)
hist(sqrt(traindataorig$Ba))
```




```
traindataorig$Fe.sqrt <- sqrt(traindataorig$Fe)
hist(sqrt(traindataorig$Fe))
```




```
# RI no transformation View(traindataorig)
```

Now that our data is ready to model, we will divide it into training (70%) and test sets (30%)

```
set.seed(21)
randomobs <- sample(seq_len(nrow(traindataorig)), size = floor(0.7 * nrow(traindataorig)))

trainnew <- traindataorig[randomobs, ]
testnew <- traindataorig[-randomobs, ]
```

1.6 Building Models

I shall use the split training dataset (70%) of observations to build the models.

1.6.1 Model 1 Multinomial Regression

As the first model, I will build the multinomial logistic regression model. All the predictors are considered. The transformed ones of **K, Ba, Fe** replace the original ones. GlassType 1 is considered the baseline.

```
library(nnet)
trainnew$GTF <- factor(trainnew$GlassType)
trainnew$out <- relevel(trainnew$GTF, ref = "1")


# View(trainnew)

model1.multi <- multinom(out ~ RI + Na + Mg + Al + Si + K.sqrt + Ca + Ba.sqrt + Fe.sqrt, data = trainnew
```

```
## # weights:  66 (50 variable)
## initial  value 266.972161
## iter  10 value 156.697075
## iter  20 value 116.876817
## iter  30 value 110.020758
## iter  40 value 109.231728
## iter  50 value 107.662616
## iter  60 value 103.867161
## iter  70 value 102.937607
## iter  80 value 102.837796
## iter  90 value 102.812516
## iter 100 value 102.611749
## final  value 102.611749
## stopped after 100 iterations
```

```
# print(model1.multi)
sum_model1 <- summary(model1.multi)
```

```
## Warning in sqrt(diag(vc)): NaNs produced
```

```
sum_model1
```

```
## Call:
## multinom(formula = out ~ RI + Na + Mg + Al + Si + K.sqrt + Ca +
##     Ba.sqrt + Fe.sqrt, data = trainnew)
##
## Coefficients:
##   (Intercept)          RI         Na          Mg         Al         Si
## 2  127.141386 248.9862786 -4.3237147  -6.3941771 -1.3898442 -5.1389309
## 3   69.434650 -44.6137239  0.5478386  -0.5551375  2.8812999 -0.1269432
## 5  -40.572900 -65.3930713  1.4341680  -1.1486837  9.9558572  1.0394699
## 6    0.592119   0.4818755 47.9497117 -24.9482272 44.9504997 -7.4759230
## 7  -40.782030  27.5347396  3.5892257  -4.6616778  0.6870919 -0.4796699
##        K.sqrt           Ca      Ba.sqrt     Fe.sqrt
## 2   -5.246227  -5.43850768   -0.5293415   0.9106844
## 3   -3.029058  -0.07664801 -223.4947929  -1.7739504
## 5   13.760704   2.34440492    9.3100173  -7.7809796
## 6 -142.326564 -12.50567591 -224.7375417 -17.0677733
```

```
## 7    8.884266  -1.23792333   11.4847699  -8.4474205
##
## Std. Errors:
##   (Intercept)         RI        Na          Mg         Al        Si
## 2  0.19455280 0.30101249 0.6489630 0.776140642 1.29586707 0.1411439
## 3  0.03583276 0.05945938 0.9475242 1.001842998 1.67838498 0.2041915
## 5  0.14675026 0.24528640 1.0250759 1.129058941 2.96411232 0.2703017
## 6  0.09796312 0.14872588 1.3441278 0.000629107 0.05512237 7.2928069
## 7  0.11517313 0.19233957 1.8883921 1.255983300 2.93481459 0.3407746
##         K.sqrt        Ca      Ba.sqrt      Fe.sqrt
## 2 2.216158e+00 0.4959267 4.477495e+00 1.191094e+00
## 3 2.531062e+00 0.6564450          NaN 2.320730e+00
## 5 3.278529e+00 0.8653744 4.779757e+00 4.128661e+00
## 6 1.757522e-08 1.0762411 1.064960e-09 4.413815e-16
## 7 3.841952e+00 0.9222100 5.187446e+00 8.722964e+00
##
## Residual Deviance: 205.2235
## AIC: 305.2235
```

```r
# Applying model to test data

testnew$GTF <- factor(testnew$GlassType)

predval <- predict(model1.multi, testnew)
confmt <- table(predval, testnew$GTF)
confm1 <- confusionMatrix(predval, testnew$GTF)
testnew$outm1 <- predval

confm1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3  5  6  7
##          1 17  3  4  0  0  0
##          2  6 15  2  1  1  0
##          3  0  0  0  0  0  0
##          5  0  0  0  2  0  0
##          6  0  1  0  0  3  0
##          7  0  1  0  0  0  9
##
## Overall Statistics
##
##                Accuracy : 0.7077
##                  95% CI : (0.5817, 0.814)
##     No Information Rate : 0.3538
##     P-Value [Acc > NIR] : 6.899e-09
##
##                   Kappa : 0.5965
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 5 Class: 6 Class: 7
## Sensitivity            0.7391   0.7500  0.00000  0.66667  0.75000   1.0000
```

```
## Specificity            0.8333   0.7778  1.00000  1.00000  0.98361   0.9821
## Pos Pred Value         0.7083   0.6000      NaN  1.00000  0.75000   0.9000
## Neg Pred Value         0.8537   0.8750  0.90769  0.98413  0.98361   1.0000
## Prevalence             0.3538   0.3077  0.09231  0.04615  0.06154   0.1385
## Detection Rate         0.2615   0.2308  0.00000  0.03077  0.04615   0.1385
## Detection Prevalence   0.3692   0.3846  0.00000  0.03077  0.06154   0.1538
## Balanced Accuracy      0.7862   0.7639  0.50000  0.83333  0.86680   0.9911
```

```
# View(testnew)

Misce1 <- 1 - sum(diag(confmt))/sum(confmt)
ztest <- sum_model1$coefficients/sum_model1$standard.errors
p <- (1 - pnorm(abs(ztest), 0, 1)) * 2
p
```

```
##    (Intercept)          RI           Na           Mg           Al
## 2 0.000000e+00 0.000000000 2.692113e-11 2.220446e-16 0.2834862470
## 3 0.000000e+00 0.000000000 5.631433e-01 5.794992e-01 0.0860322094
## 5 0.000000e+00 0.000000000 1.617876e-01 3.089720e-01 0.0007828201
## 6 1.500549e-09 0.001195195 0.000000e+00 0.000000e+00 0.0000000000
## 7 0.000000e+00 0.000000000 5.734417e-02 2.059726e-04 0.8148936368
##             Si       K.sqrt          Ca    Ba.sqrt    Fe.sqrt
## 2 0.0000000000 1.792022e-02 0.000000000 0.90589121 0.44452299
## 3 0.5341476340 2.314026e-01 0.907048440        NaN 0.44463289
## 5 0.0001202622 2.702129e-05 0.006746149 0.05143873 0.05948044
## 6 0.3053116950 0.000000e+00 0.000000000 0.00000000 0.00000000
## 7 0.1592533723 2.075370e-02 0.179484412 0.02683190 0.33283879
```

```
model1cf_p1 <- as.data.frame(confm1$overall)
model1cf_p2 <- as.data.frame(confm1$byClass)
colnames(model1cf_p1) <- "Model1"
colnames(model1cf_p2) <- "Model1"

coefficients(model1.multi)
```

```
##   (Intercept)          RI         Na          Mg         Al         Si
## 2  127.141386 248.9862786 -4.3237147  -6.3941771 -1.3898442 -5.1389309
## 3   69.434650 -44.6137239  0.5478386  -0.5551375  2.8812999 -0.1269432
## 5  -40.572900 -65.3930713  1.4341680  -1.1486837  9.9558572  1.0394699
## 6    0.592119   0.4818755 47.9497117 -24.9482272 44.9504997 -7.4759230
## 7  -40.782030  27.5347396  3.5892257  -4.6616778  0.6870919 -0.4796699
##        K.sqrt          Ca      Ba.sqrt     Fe.sqrt
## 2   -5.246227  -5.43850768   -0.5293415   0.9106844
## 3   -3.029058  -0.07664801 -223.4947929  -1.7739504
## 5   13.760704   2.34440492    9.3100173  -7.7809796
## 6 -142.326564 -12.50567591 -224.7375417 -17.0677733
## 7    8.884266  -1.23792333   11.4847699  -8.4474205
```

```
# Finding AIC and BIC

aicm1 <- AIC(model1.multi)
bicm1 <- BIC(model1.multi)
```

Glass Type 1 is the baseline reference level.

It is observed here that RI Refractive Index has a positive and very high coefficient for GlassType 2, where as it is negatively impacting Type 3 and 5. However its significance is high as seen from p values.

The presence of Na, sodium is not of much significance for Type in identifying any of the Glass Type except for Type 3,5,7. It has high significane for Type 3 (low coefficient)and Type 6(positive high coefficeint) glass.

Mg, Magnesium is observed to have high significance for Type 2 ,6,7, Has negative coefficeints.

Al,Aluminium has very high coefficeint for Type 6 and high significance for Type 5, 6 Si hold statistical significance for Type 2, 5

Potassium K, Barium (high coefficient), Calcium hold significance for Type 6

Potassium also significant in identifying Type 2,7

The accuracy of the model is 70%

```
par(mfrow = c(1, 2))

pander(ftable(confm1$table), caption = "Confusion Matrix :Model 1 Multinomial Logistic Regression")
```

Table 7: Confusion Matrix :Model 1 Multinomial Logistic Regression

	"Reference"	"1"	"2"	"3"	"5"	"6"	"7"
"Prediction"							
"1"		17	3	4	0	0	0
"2"		6	15	2	1	1	0
"3"		0	0	0	0	0	0
"5"		0	0	0	2	0	0
"6"		0	1	0	0	3	0
"7"		0	1	0	0	0	9

1.6.2 Model 2 Random Forest

The "caret" package train function is used with method of "random forest"" . A 5 fold cross validation is used. From the summary we see that for 2 mtry (number of variables used), we get the highest accuracy of 86%

From the plot of the fit, we observe how the accuracy decreases as the predictors increase..

The top impacting predictor variables that give the best performance (highest accuracy are plotted in the Variable Importance Plot below. The plot shows the importance per Glass Type)category

```
model2.rf <- train(out ~ RI + Na + Mg + Al + Si + K.sqrt + Ca + Ba.sqrt + Fe.sqrt, data = trainnew, metl
    number = 5), prox = TRUE, importance = TRUE, allowParallel = TRUE)
```

```
## Warning: package 'randomForest' was built under R version 3.3.2
```

```
# show the model summary
sum_model2 <- summary(model2.rf)
print(model2.rf)
```

```
## Random Forest
##
## 149 samples
##   9 predictor
##   6 classes: '1', '2', '3', '5', '6', '7'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 120, 120, 120, 118, 118
## Resampling results across tuning parameters:
```

```
##
##   mtry  Accuracy   Kappa
##   2     0.7730812  0.6748534
##   5     0.7325918  0.6254209
##   9     0.6916574  0.5681717
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
# Crossvalidating for test model
predval <- predict(model2.rf, testnew)
confmt <- table(predval, testnew$GTF)
confm2 <- confusionMatrix(predval, testnew$GTF)
testnew$outm2 <- predval

confm2
```

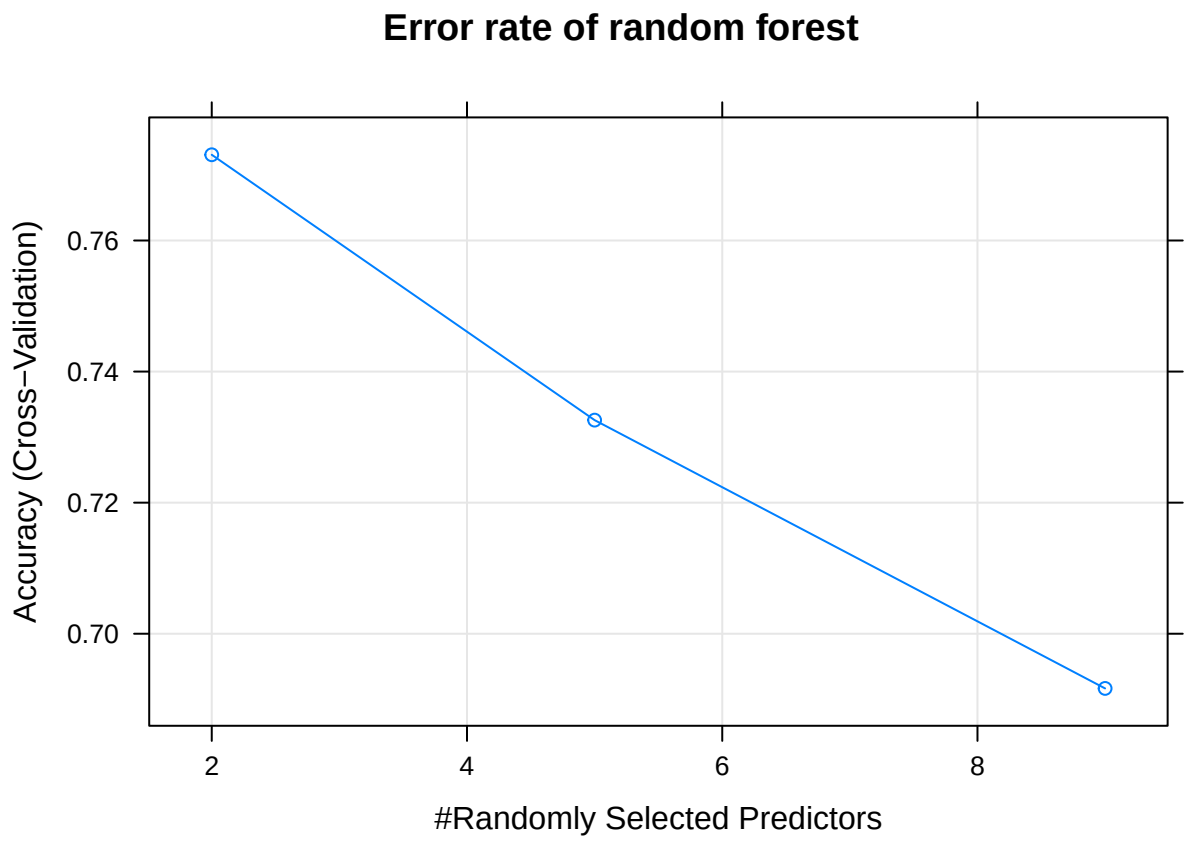
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3  5  6  7
##          1 22  1  4  0  0  0
##          2  1 19  1  0  0  2
##          3  0  0  1  0  0  0
##          5  0  0  0  3  0  0
##          6  0  0  0  0  4  0
##          7  0  0  0  0  0  7
##
## Overall Statistics
##
##                Accuracy : 0.8615
##                  95% CI : (0.7534, 0.9347)
##     No Information Rate : 0.3538
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8082
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 5 Class: 6 Class: 7
## Sensitivity            0.9565   0.9500  0.16667  1.00000  1.00000   0.7778
## Specificity            0.8810   0.9111  1.00000  1.00000  1.00000   1.0000
## Pos Pred Value         0.8148   0.8261  1.00000  1.00000  1.00000   1.0000
## Neg Pred Value         0.9737   0.9762  0.92187  1.00000  1.00000   0.9655
## Prevalence             0.3538   0.3077  0.09231  0.04615  0.06154   0.1385
## Detection Rate         0.3385   0.2923  0.01538  0.04615  0.06154   0.1077
## Detection Prevalence   0.4154   0.3538  0.01538  0.04615  0.06154   0.1077
## Balanced Accuracy      0.9187   0.9306  0.58333  1.00000  1.00000   0.8889
```

```
# View(testnew)

Misce2 <- 1 - sum(diag(confmt))/sum(confmt)

model2cf_p1 <- as.data.frame(confm2$overall)
```

```r
model2cf_p2 <- as.data.frame(confm2$byClass)
colnames(model2cf_p1) <- "Model2"
colnames(model2cf_p1) <- "Model2"

coefficients(model2.rf)
```

```
## NULL
```

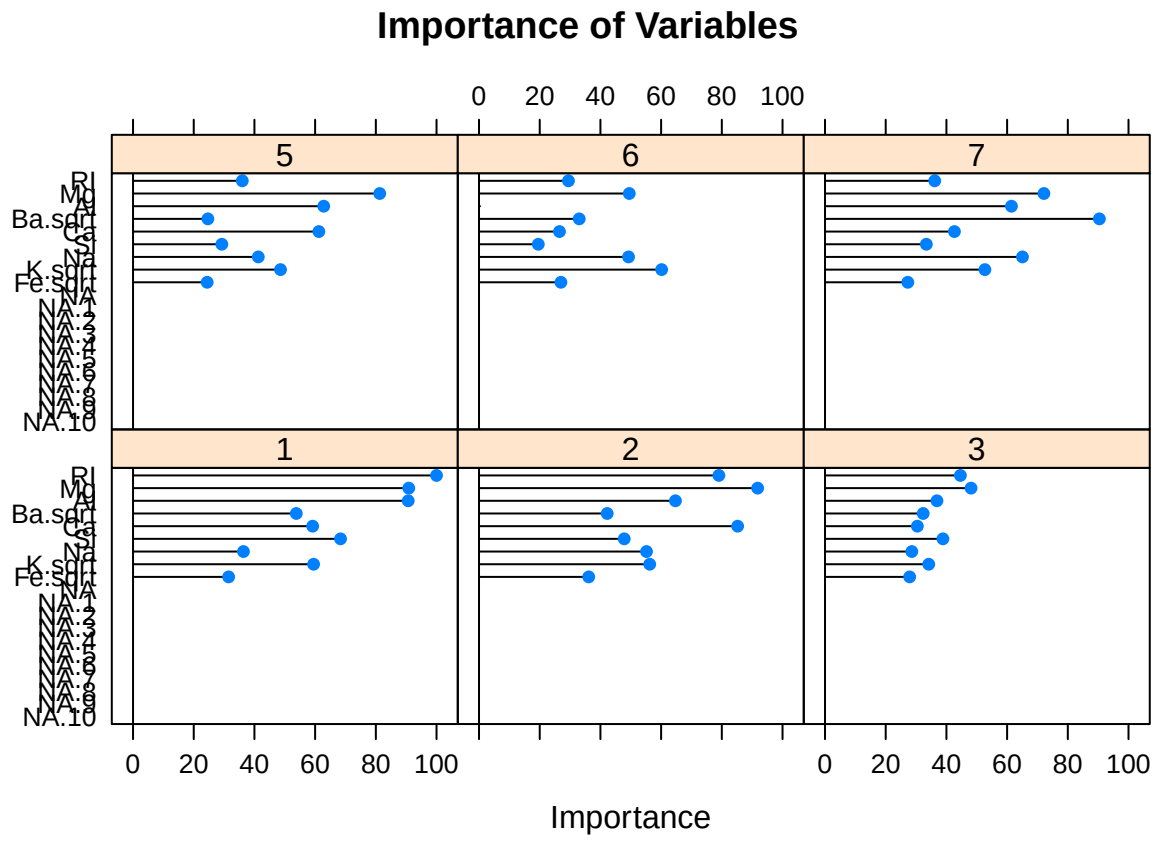
```r
plot(model2.rf, main = "Error rate of random forest")
```




```r
## variable importance
rfImp <- varImp(model2.rf)
rfImp
```

```
## rf variable importance
##
##   variables are sorted by maximum importance across the classes
##              1     2     3     5     6     7
## RI      100.00 79.03 44.61 36.00 29.47 36.15
## Mg       90.85 91.80 48.12 81.29 49.51 72.13
## Al       90.67 64.72 36.88 62.83  0.00 61.43
## Ba.sqrt  53.79 42.23 32.34 24.67 33.02 90.40
## Ca       59.19 85.21 30.43 61.22 26.51 42.64
## Si       68.37 47.85 38.85 29.26 19.53 33.39
## Na       36.38 55.19 28.60 41.27 49.26 65.05
## K.sqrt   59.50 56.25 34.18 48.59 60.17 52.68
## Fe.sqrt  31.51 36.15 27.90 24.43 26.98 27.28
```

```
plot(rfImp, top = 20, main = "Importance of Variables")
```




```
pander(ftable(confm2$table), caption = "Confusion Matrix :Model 2 Random Forest Model")
```

Table 8: Confusion Matrix :Model 2 Random Forest Model

	“Reference”	“1”	“2”	“3”	“5”	“6”	“7”
“Prediction”							
“1”		22	1	4	0	0	0
“2”		1	19	1	0	0	2
“3”		0	0	1	0	0	0
“5”		0	0	0	3	0	0
“6”		0	0	0	0	4	0
“7”		0	0	0	0	0	7

The variable importance plot is a critical output of the random forest algorithm. For each variable in your matrix it tells you how important that variable is in classifying the data. The plot shows each variable on the y-axis, and their importance on the x-axis. They are ordered top-to-bottom as most- to least-important.

Although the importance of variables is varying across the glass types , it is observed that Mg , Magnesium is the top player among all (topmost important for Type 5,2,3 and second topmost for 1,6,7)

Similarly, Refreactive Index helps in identification for window and vehicle appplication glasses. Potassium seems to be of mediocre importance for all types.

Aluminium has no importance for tableware glassbut quite important for Type 5 container, and in Type 1,2,3 ie window/vehicle glasses

Si is of impotance in window glasses.

1.6.3 Model 3 Conditional Inference Tree

This model will same predictor set as the earlier two models. Party package ctree function is used for this model.

```
library(party)
```

```
## Warning: package 'party' was built under R version 3.3.3
```

```
## Warning: package 'mvtnorm' was built under R version 3.3.2
```

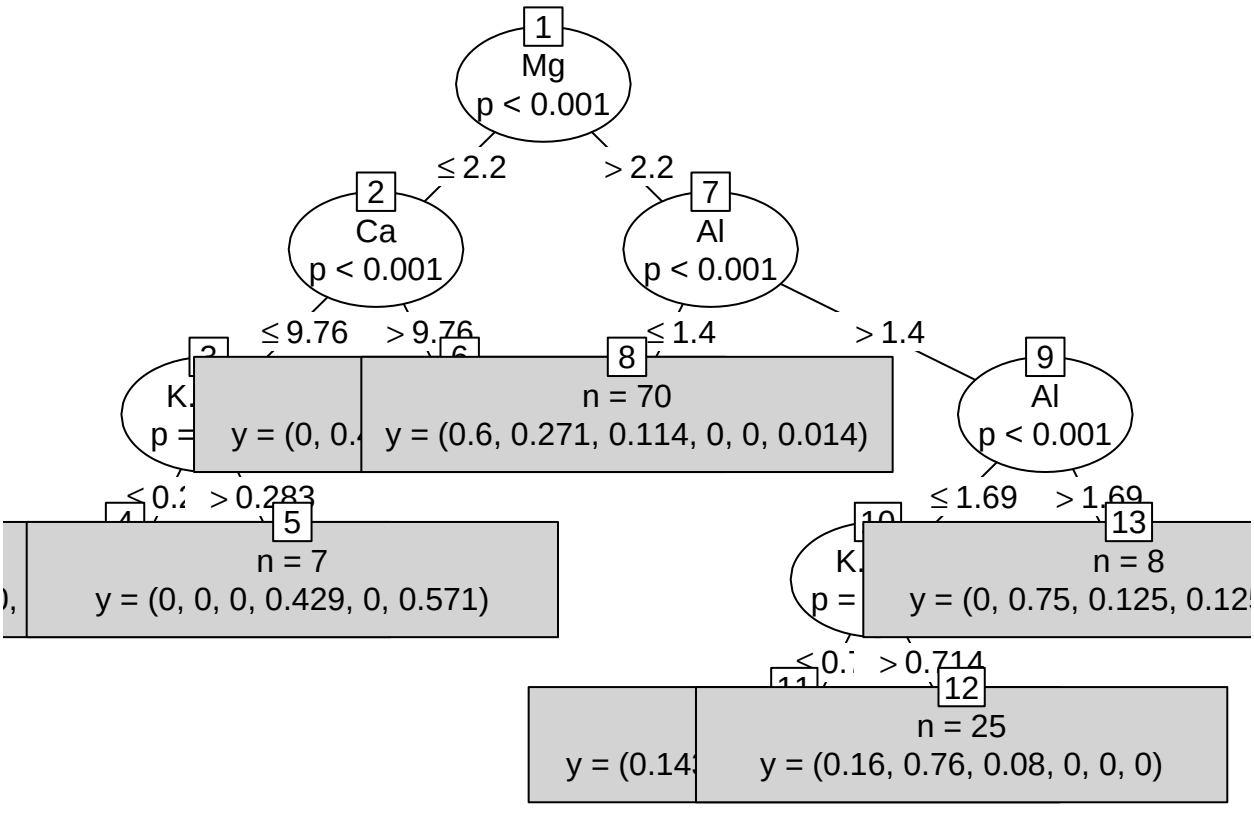
```
## Warning: package 'modeltools' was built under R version 3.3.2
```

```
## Warning: package 'strucchange' was built under R version 3.3.3
```

```
## Warning: package 'zoo' was built under R version 3.3.3
```

```
## Warning: package 'sandwich' was built under R version 3.3.3
```

```
Fmla <- out ~ RI + Na + Mg + Al + Si + K.sqrt + Ca + Ba.sqrt + Fe.sqrt
model3.tree <- ctree(Fmla, data = trainnew)
plot(model3.tree, type = "simple")
```




```
# show the model summary
sum_model3 <- summary(model3.tree)
sum_model3
```

```
##      Length       Class        Mode
```

```
##          1 BinaryTree         S4
```

```
print(model3.tree)
```

```
##
##   Conditional inference tree with 7 terminal nodes
##
## Response:  out
## Inputs:  RI, Na, Mg, Al, Si, K.sqrt, Ca, Ba.sqrt, Fe.sqrt
## Number of observations:  149
##
## 1) Mg <= 2.2; criterion = 1, statistic = 98.709
##   2) Ca <= 9.76; criterion = 0.999, statistic = 22.298
##     3) K.sqrt <= 0.2828427; criterion = 0.99, statistic = 13.604
##       4)*  weights = 16
##     3) K.sqrt > 0.2828427
##       5)*  weights = 7
##   2) Ca > 9.76
##     6)*  weights = 16
## 1) Mg > 2.2
##   7) Al <= 1.4; criterion = 1, statistic = 97.618
##     8)*  weights = 70
##   7) Al > 1.4
##     9) Al <= 1.69; criterion = 1, statistic = 37.965
##       10) K.sqrt <= 0.7141428; criterion = 0.998, statistic = 18.988
##         11)*  weights = 7
##       10) K.sqrt > 0.7141428
##         12)*  weights = 25
##     9) Al > 1.69
##       13)*  weights = 8
```

```
# Crossvalidating for test model
predval <- predict(model3.tree, testnew)
confmt <- table(predval, testnew$GTF)
confm3 <- confusionMatrix(predval, testnew$GTF)
testnew$outm3 <- predval

confm3
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3  5  6  7
##          1 21  3  5  0  1  0
##          2  2 16  1  3  1  2
##          3  0  0  0  0  0  0
##          5  0  0  0  0  0  0
##          6  0  0  0  0  0  0
##          7  0  1  0  0  2  7
##
## Overall Statistics
##
##                Accuracy : 0.6769
##                  95% CI : (0.5495, 0.7877)
##     No Information Rate : 0.3538
##     P-Value [Acc > NIR] : 1.184e-07
```

```
##
##                   Kappa : 0.5365
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 5 Class: 6 Class: 7
## Sensitivity            0.9130   0.8000  0.00000  0.00000  0.00000   0.7778
## Specificity            0.7857   0.8000  1.00000  1.00000  1.00000   0.9464
## Pos Pred Value         0.7000   0.6400      NaN      NaN      NaN   0.7000
## Neg Pred Value         0.9429   0.9000  0.90769  0.95385  0.93846   0.9636
## Prevalence             0.3538   0.3077  0.09231  0.04615  0.06154   0.1385
## Detection Rate         0.3231   0.2462  0.00000  0.00000  0.00000   0.1077
## Detection Prevalence   0.4615   0.3846  0.00000  0.00000  0.00000   0.1538
## Balanced Accuracy      0.8494   0.8000  0.50000  0.50000  0.50000   0.8621
```

```r
# View(testnew)

Misce3 <- 1 - sum(diag(confmt))/sum(confmt)

model3cf_p1 <- as.data.frame(confm3$overall)
model3cf_p2 <- as.data.frame(confm3$byClass)
colnames(model3cf_p1) <- "Model3"
colnames(model3cf_p2) <- "Model3"

pander(ftable(confm3$table), caption = "Confusion Matrix : Model 3 Conditional Inference Tree Model")
```

Table 9: Confusion Matrix : Model 3 Conditional Inference Tree Model

“Reference”	“1”	“2”	“3”	“5”	“6”	“7”
“Prediction”						
“1”	21	3	5	0	1	0
“2”	2	16	1	3	1	2
“3”	0	0	0	0	0	0
“5”	0	0	0	0	0	0
“6”	0	0	0	0	0	0
“7”	0	1	0	0	2	7

The dataset is here partitioned into two subsets based on Mg content for values > 2.2 and <= 2.2.

Either of the branches are further split based on other chemical constituents. The greater than 2.2 Mg content group is further analyzed for Al , Aluminium content based on Al avlues of <=1.4 or >1.4.

The key point is that every record in the dataset is ultimately assigned to one of the terminal nodes of this tree (the “leaves,” numbered 4,5,8,11,12,13 and 13). The numbers associated with these nodes gives their size and the fraction of each glass type which may be viewed as an estimate of the conditional probability.

Although the model determines the important predictors similar to Randome forest , for Mg , Al, it is not upto mark with only 67% accuracy.

1.6.4 Model 4 Linear Discriminant Analysis

I will proceed with LDA model as the fourth model for classification of glasstypes

```r
model4.lda <- train(out ~ RI + Na + Mg + Al + Si + K.sqrt + Ca + Ba.sqrt + Fe.sqrt, data = trainnew, me

sum_model4 <- summary(model4.lda)



# Crossvalidating for test model
predval <- predict(model4.lda, testnew)
confmt <- table(predval, testnew$GTF)
confm4 <- confusionMatrix(predval, testnew$GTF)
testnew$outm <- predval

confm4
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3  5  6  7
##          1 10  3  5  0  0  0
##          2 13 16  1  1  1  1
##          3  0  0  0  0  0  0
##          5  0  0  0  1  0  0
##          6  0  1  0  1  3  0
##          7  0  0  0  0  0  8
##
## Overall Statistics
##
##                Accuracy : 0.5846
##                  95% CI : (0.4556, 0.7056)
##     No Information Rate : 0.3538
##     P-Value [Acc > NIR] : 0.0001239
##
##                   Kappa : 0.4257
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 5 Class: 6 Class: 7
## Sensitivity            0.4348   0.8000  0.00000  0.33333  0.75000   0.8889
## Specificity            0.8095   0.6222  1.00000  1.00000  0.96721   1.0000
## Pos Pred Value         0.5556   0.4848      NaN  1.00000  0.60000   1.0000
## Neg Pred Value         0.7234   0.8750  0.90769  0.96875  0.98333   0.9825
## Prevalence             0.3538   0.3077  0.09231  0.04615  0.06154   0.1385
## Detection Rate         0.1538   0.2462  0.00000  0.01538  0.04615   0.1231
## Detection Prevalence   0.2769   0.5077  0.00000  0.01538  0.07692   0.1231
## Balanced Accuracy      0.6222   0.7111  0.50000  0.66667  0.85861   0.9444
```

```r
# View(testnew)

Misce4 <- 1 - sum(diag(confmt))/sum(confmt)

model4cf_p1 <- as.data.frame(confm4$overall)
model4cf_p2 <- as.data.frame(confm4$byClass)
colnames(model4cf_p1) <- "Model4"
```

```
colnames(model4cf_p2) <- "Model4"

pander(ftable(confm4$table), caption = "Confusion Matrix : Model 4 Linear Discriminant Analysis")
```

Table 10: Confusion Matrix : Model 4 Linear Discriminant Analysis

	"Reference"	"1"	"2"	"3"	"5"	"6"	"7"
"Prediction"							
"1"		10	3	5	0	0	0
"2"		13	16	1	1	1	1
"3"		0	0	0	0	0	0
"5"		0	0	0	1	0	0
"6"		0	1	0	1	3	0
"7"		0	0	0	0	0	8

```
# multiclass.roc(testnew$GTF, predict(model4.lda, testnew, type = response))
```

LDA seems to be giving the worst accuracy of only 58%.

1.7 Model Selection and Inference:

From the four models derived above, we look at the performance of each of these through cross validation, with respect to the Accuracy in classification. Form the comparison table below we see that highest Accuracy, least misclassification error is delivered by Random Forest Model is the highest.

The Factors most impacting the GlassType are Magnesium, Calcium, Aluminium and Refractive Index, Potassium although others are significant to smaller extent.

Since there is no evaluation dataset, the test (30% of randomly selected observations from original data) are written to *predicted_glasstypes.csv*

Note : outm1,outm2,outm3,outm4 are the predictions of each Model1 , Model2, Model3 , Model4

GTF : Factored GlassType

GlassType : Response Variable

Confusion Matrix Metrics For All Models

```
cfmatmetricsdf <- cbind(model1cf_p1, model2cf_p1, model3cf_p1, model4cf_p1)
kable(cfmatmetricsdf, caption = "Confusion Matrix Metrics For All Models")
```

Table 11: Confusion Matrix Metrics For All Models

	Model1	Model2	Model3	Model4
Accuracy	0.7076923	0.8615385	0.6769231	0.5846154
Kappa	0.5965371	0.8081967	0.5365025	0.4257199
AccuracyLower	0.5817413	0.7533617	0.5494504	0.4556447
AccuracyUpper	0.8139710	0.9346804	0.7876724	0.7056180
AccuracyNull	0.3538462	0.3538462	0.3538462	0.3538462
AccuracyPValue	0.0000000	0.0000000	0.0000001	0.0001239
McnemarPValue	NaN	NaN	NaN	NaN

```
# cfmatmetricsdf2 <- cbind(model1cf_p2,model2cf_p2,model3cf_p2,model4cf_p2)
# kable(cfmatmetricsdf2,caption='Confusion Matrix Metrics For All Models')
```

Comparison For All Models

```
# Misclassification Error For All Models
MCEAll <- rbind(Misce1, Misce2, Misce3, Misce4) %>% round(2)


comptable <- cbind(MCEAll)

rownames(comptable) <- c("Model 1", "Model 2", "Model 3", "Model 4")
colnames(comptable) <- c("MCE")

pander(comptable, caption = "Model Comparison: MisClassification Error")
```

Table 12: Model Comparison: MisClassification Error

	MCE
Model 1	0.29
Model 2	0.14
Model 3	0.32
Model 4	0.42

MCE

```
write.csv(testnew, "predicted_glasstypes.csv")
```

1.7.1 Reference :

http://www.asdlib.org/onlineArticles/elabware/thompson/Glass/Glass(RI)PFaculty.pdf

https://en.wikipedia.org/wiki/Glass

http://www.cmog.org/article/chemistry-glass

http://www.britglass.org.uk/alkali-barium-silicate-glass

1.7.1.1 *The CrystalGazer's Visions Have Flashed !!*