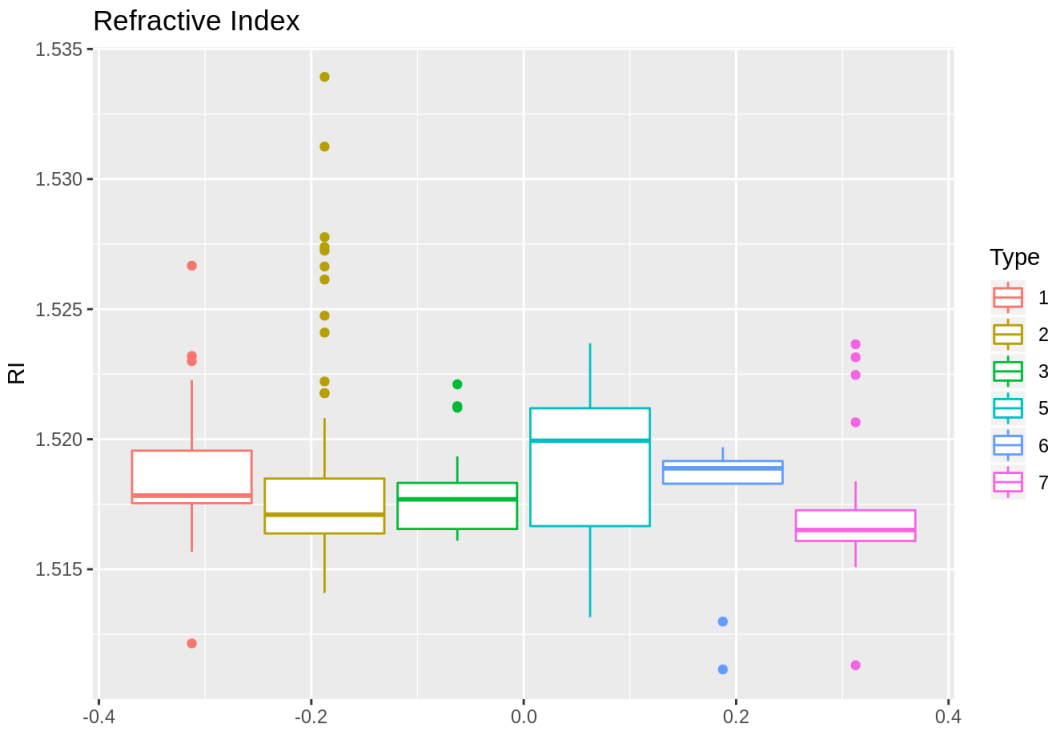
Classification

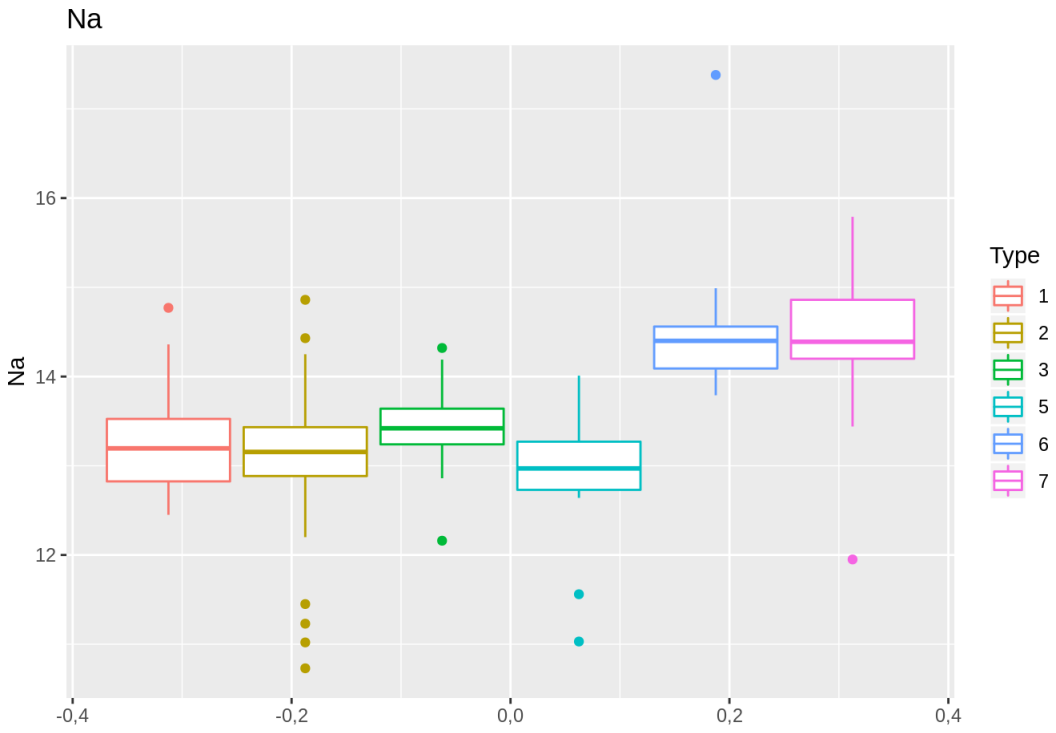
```
library(mlbench)
data(Glass)
library(ggplot2)
library(class)
library(caret)
library(e1071)
```

1. Boxplots

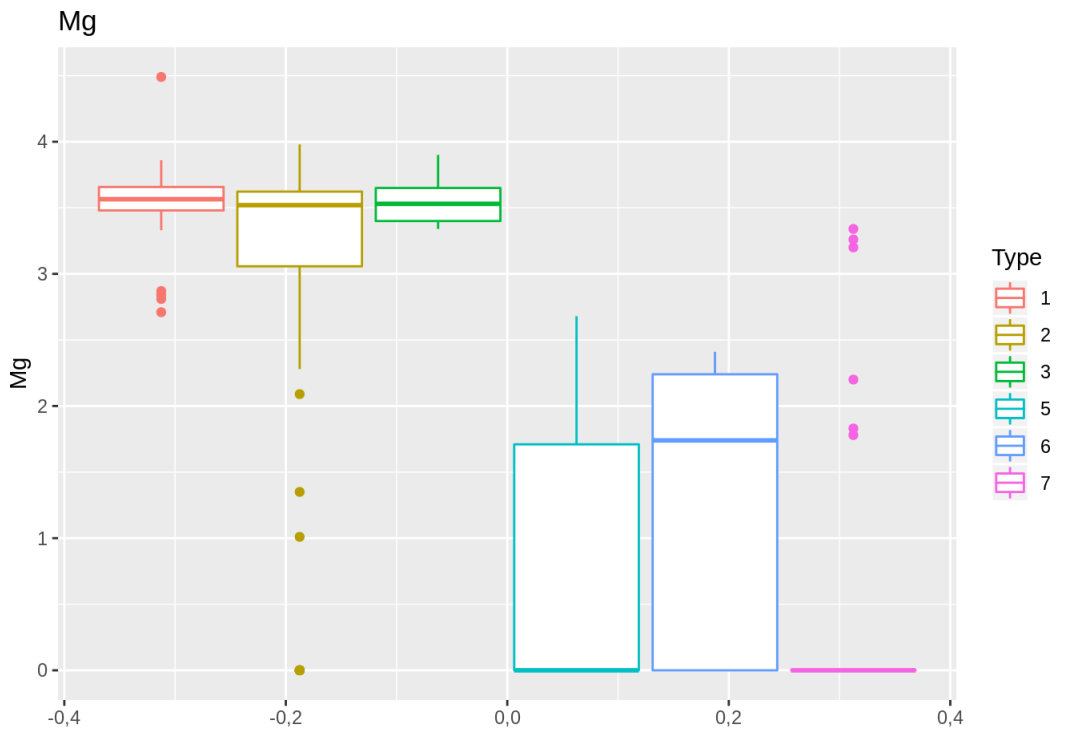
```
qplot(data = Glass, y = RI, color = Type, geom = "boxplot", main = "Refractive Index")
```



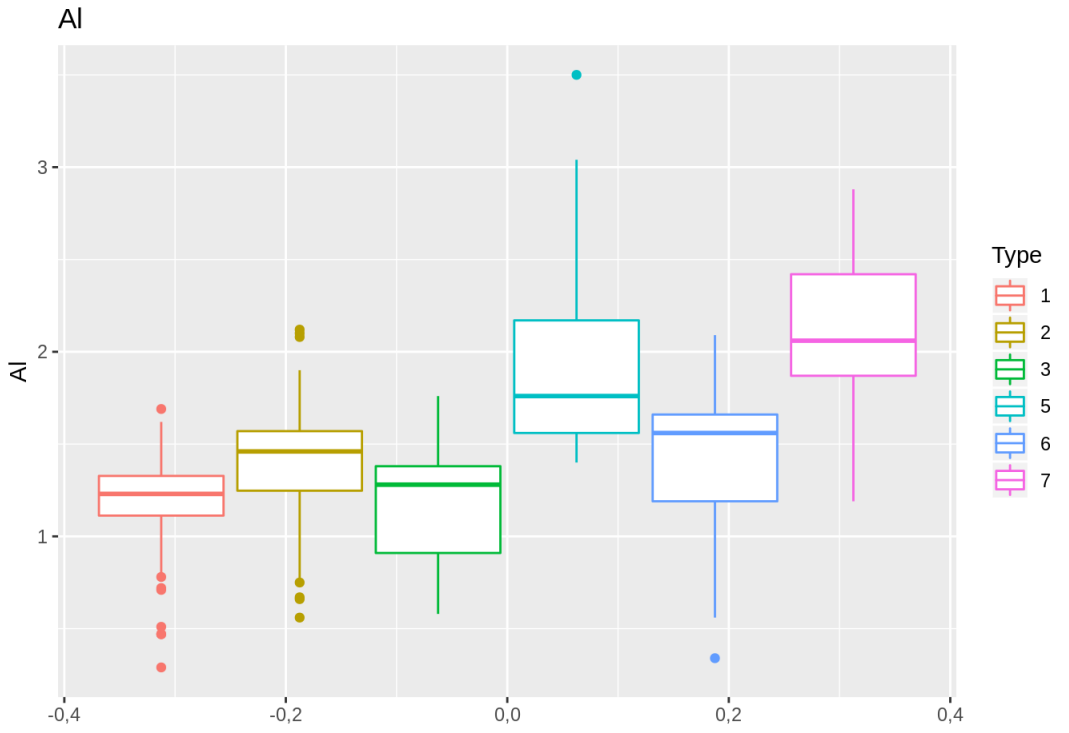

```
qplot(data = Glass, y = Na, color = Type, geom = "boxplot", main = "Na")
```



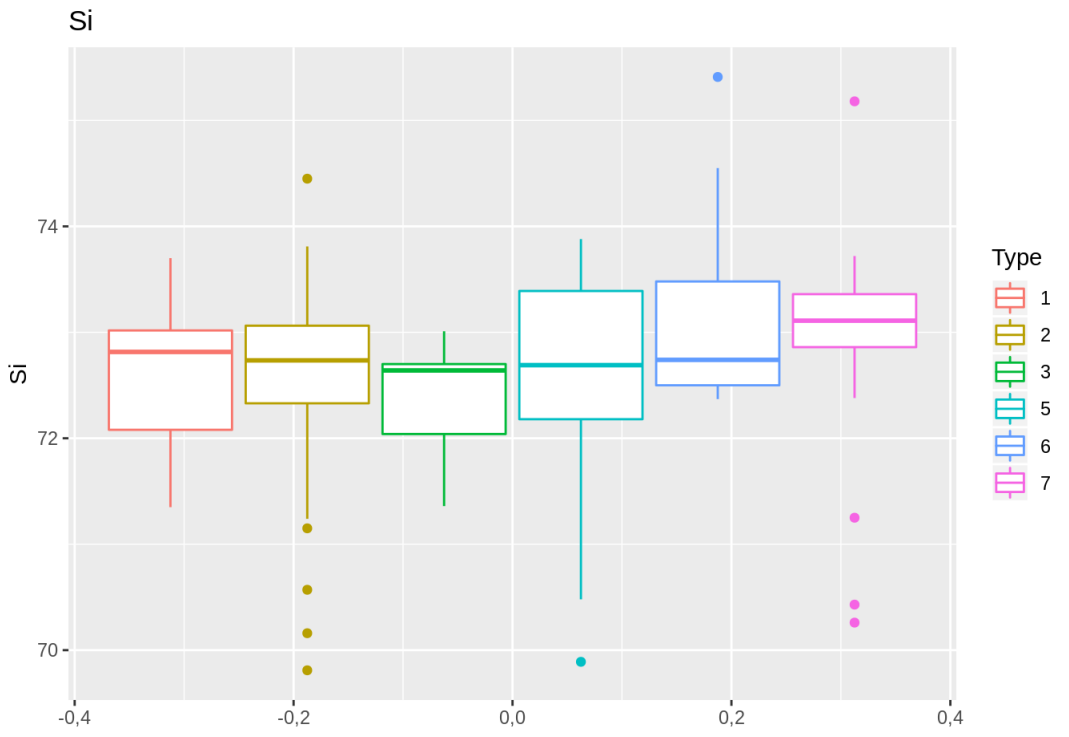

```
qplot(data = Glass, y = Mg, color = Type, geom = "boxplot", main = "Mg")
```



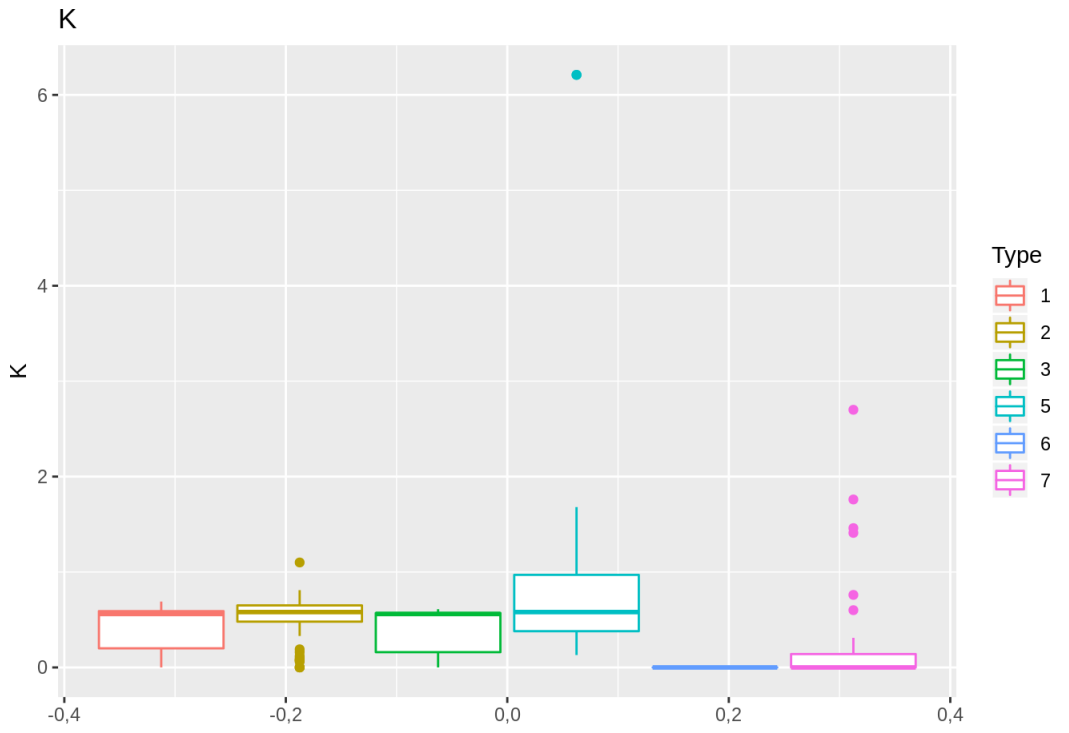

```
qplot(data = Glass, y = Al, color = Type, geom = "boxplot", main = "Al")
```



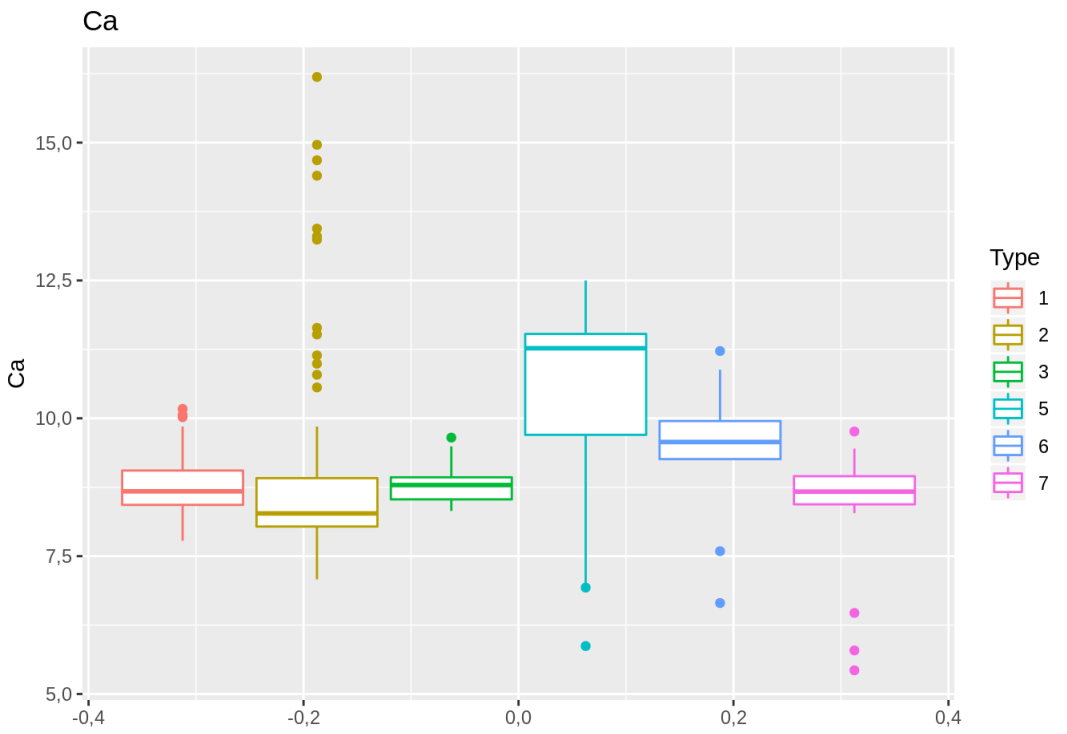

```
qplot(data = Glass, y = Si, color = Type, geom = "boxplot", main = "Si")
```



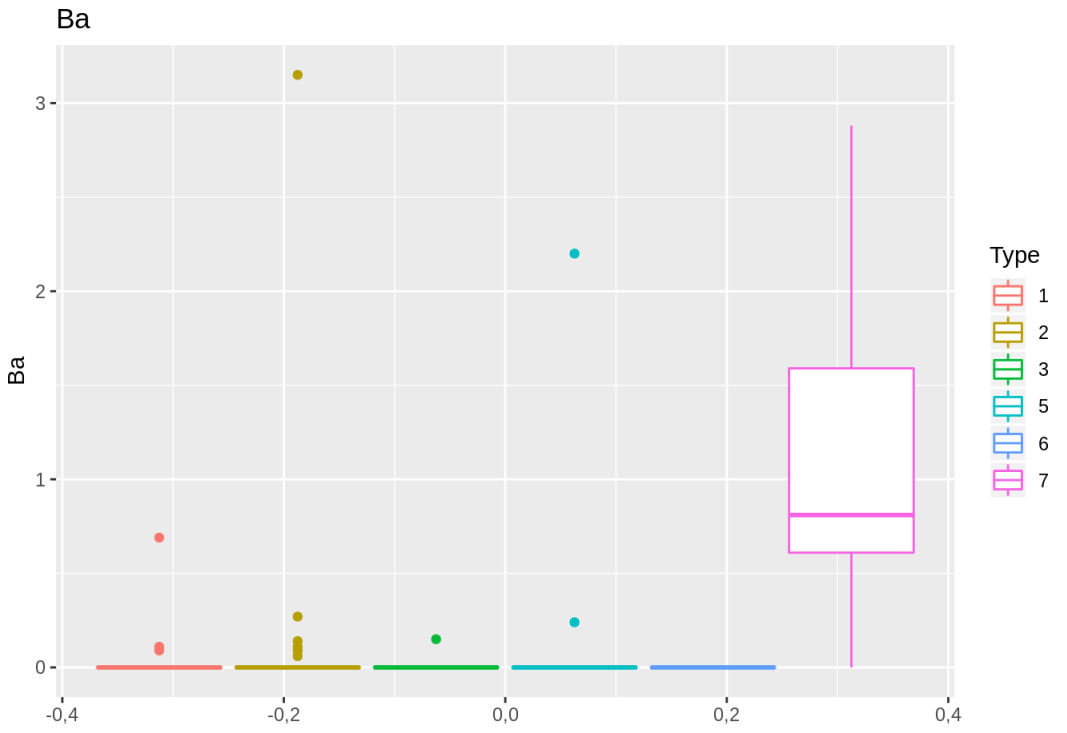

```
qplot(data = Glass, y = K, color = Type, geom = "boxplot", main = "K")
```



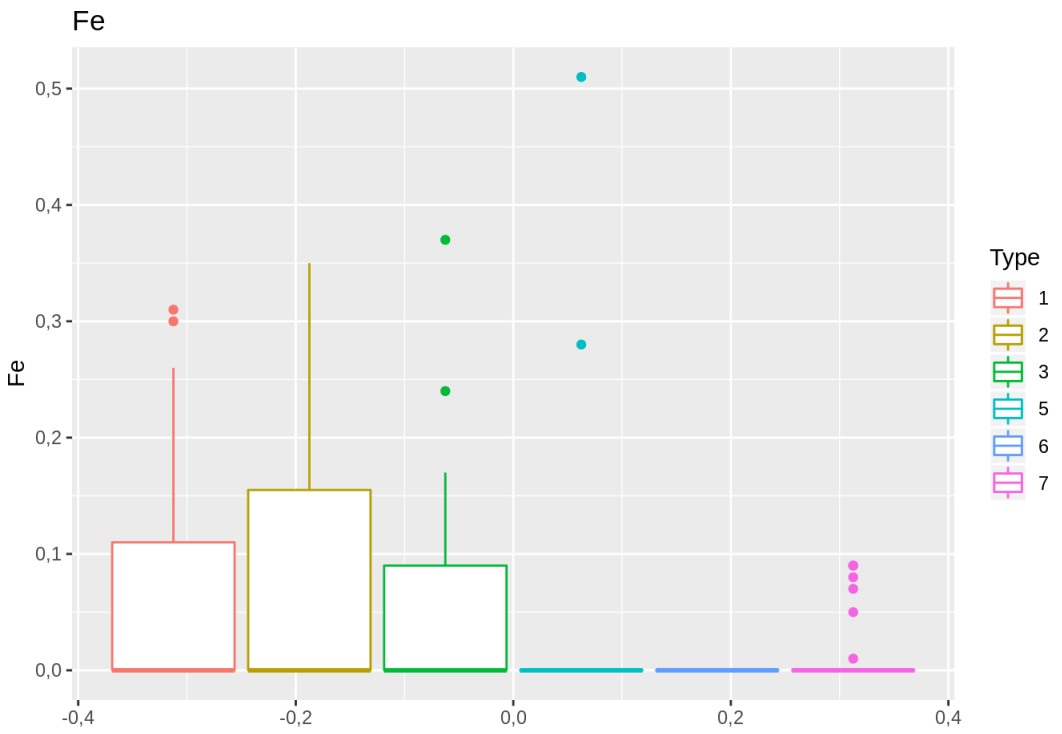

```
qplot(data = Glass, y = Ca, color = Type, geom = "boxplot", main = "Ca")
```




```
qplot(data = Glass, y = Ba, color = Type, geom = "boxplot", main = "Ba")
```




```
qplot(data = Glass, y = Fe, color = Type, geom = "boxplot", main = "Fe")
```



2. KNN

a. Al

```
set.seed(42)
# Stratified sampling
test <- createDataPartition(Glass$Type, p=0.25, list = FALSE)

pred_knn <- knn(train = Glass[-test, "Al", drop = FALSE],
                test = Glass[test, "Al", drop = FALSE],

                cl = Glass[-test, "Type"],
                k = 1)

table(pred_knn, Real = Glass[test, "Type"])
```

```
##         Real
## pred_knn  1  2  3  5  6  7
##        1 11  4  3  1  1  1
##        2  4  9  2  1  2  3
##        3  3  5  0  0  0  0
##        5  0  0  0  1  0  0
##        6  0  1  0  1  0  1
##        7  0  0  0  0  0  3
```

```
mean(pred_knn == Glass[test, "Type"])
```

```
## [1] 0,4210526
```

```
set.seed(42)
pred_knn <- knn(train = Glass[-test, "Al", drop = FALSE],
                test = Glass[test, "Al", drop = FALSE],

                cl = Glass[-test, "Type"],
                k = 10)

table(pred_knn, Real = Glass[test, "Type"])
```

```
##         Real
## pred_knn  1  2  3  5  6  7
##        1 12  5  3  0  2  0
##        2  6 12  2  3  1  4
##        3  0  0  0  0  0  0
##        5  0  0  0  0  0  0
##        6  0  0  0  0  0  0
##        7  0  2  0  1  0  4
```

```
mean(pred_knn == Glass[test, "Type"])
```

```
## [1] 0,4912281
```

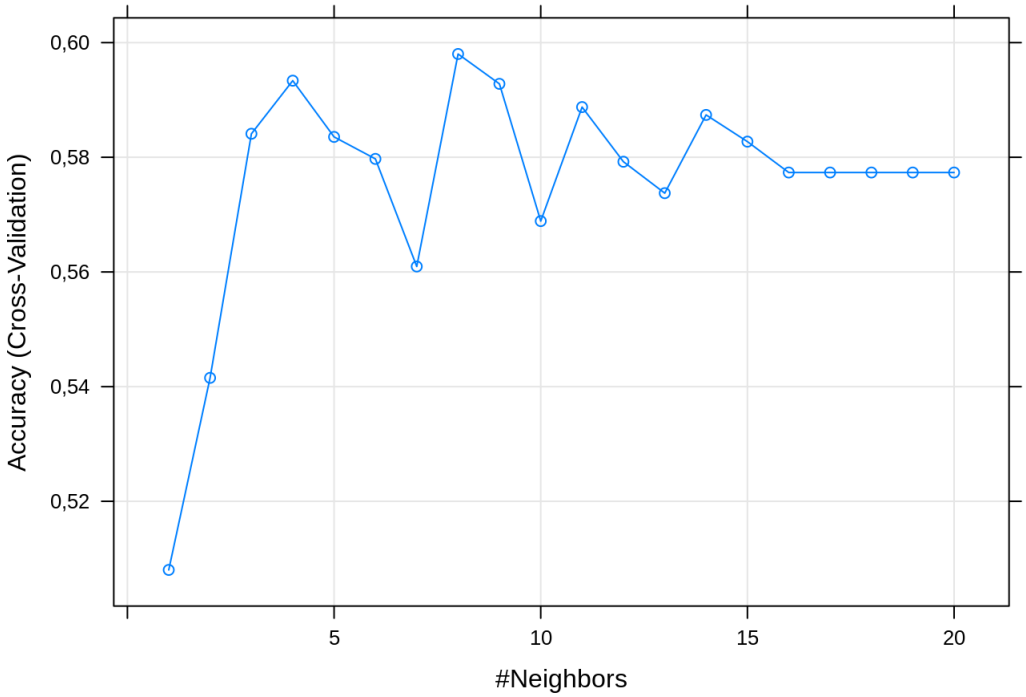
k-fold cross val

```
set.seed(42)

trControl <- trainControl(method  = "cv",
                          number  = 12)
fit <- train(Type ~ Al,
             method     = "knn",
             tuneGrid   = expand.grid(k = 1:20),
             trControl  = trControl,
             metric     = "Accuracy",
             data       = Glass)
```

```
fit
```

```
## k-Nearest Neighbors
##
## 214 samples
##   1 predictor
##   6 classes: '1', '2', '3', '5', '6', '7'
##
## No pre-processing
## Resampling: Cross-Validated (12 fold)
## Summary of sample sizes: 198, 197, 197, 195, 196, 197, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0,5080320  0,3176503
##    2  0,5415287  0,3638706
##    3  0,5841037  0,4130965
##    4  0,5933630  0,4154579
##    5  0,5835537  0,3973475
##    6  0,5797178  0,3967636
##    7  0,5609556  0,3661789
##    8  0,5980159  0,4180761
##    9  0,5928076  0,4153286
##   10  0,5688711  0,3770669
##   11  0,5887567  0,4025011
##   12  0,5792251  0,3875212
##   13  0,5737444  0,3792962
##   14  0,5873896  0,3993399
##   15  0,5827313  0,3934768
##   16  0,5773420  0,3856693
##   17  0,5773420  0,3856693
##   18  0,5773420  0,3856693
##   19  0,5773420  0,3856693
##   20  0,5773420  0,3860041
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 8.
```

```
plot(fit)
```



b. RI, Al, Si

```
set.seed(42)
pred_knn <- knn(train = Glass[-test, c("RI","Al","Si")],
                test = Glass[test, c("RI","Al","Si")],

                cl = Glass[-test, "Type"],
                k = 1)

table(pred_knn, Real = Glass[test, "Type"])
```

```
##         Real
## pred_knn  1  2  3  5  6  7
##        1 12  1  1  1  1  1
##        2  4 14  1  2  1  3
##        3  1  3  2  0  0  0
##        5  0  1  0  1  0  0
##        6  1  0  1  0  0  0
##        7  0  0  0  0  1  4
```

```
mean(pred_knn == Glass[test, "Type"])
```

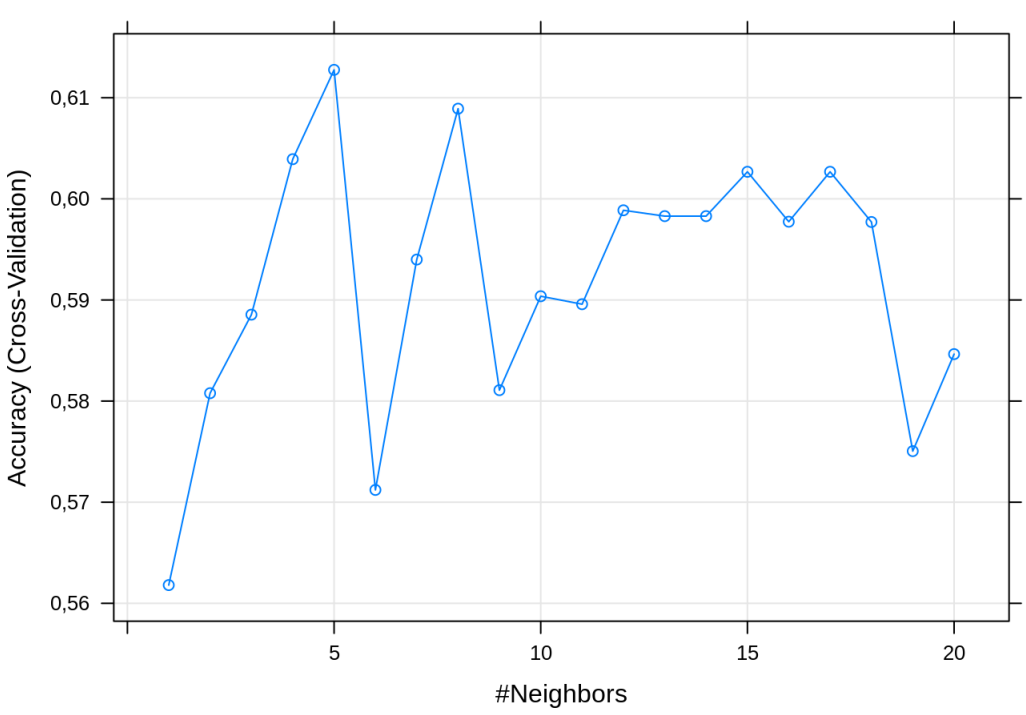
```
## [1] 0,5789474
```

```
set.seed(42)

trControl <- trainControl(method  = "cv",
                          number  = 12)

fit <- train(Type ~ RI + Al + Si,
             method     = "knn",
             tuneGrid   = expand.grid(k = 1:20),
             trControl  = trControl,
             metric     = "Accuracy",
             data       = Glass)
```

```
fit
```

```
## k-Nearest Neighbors
##
## 214 samples
##   3 predictor
##   6 classes: '1', '2', '3', '5', '6', '7'
##
## No pre-processing
## Resampling: Cross-Validated (12 fold)
## Summary of sample sizes: 198, 197, 197, 195, 196, 197, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0,5617995  0,4099565
##    2  0,5807784  0,4319915
##    3  0,5885470  0,4304192
##    4  0,6039212  0,4436183
##    5  0,6127558  0,4529520
##    6  0,5712128  0,3939731
##    7  0,5939937  0,4269579
##    8  0,6089145  0,4416142
##    9  0,5810794  0,4011413
##   10  0,5903674  0,4070353
##   11  0,5895790  0,4017001
##   12  0,5988670  0,4161133
##   13  0,5982883  0,4151980
##   14  0,5982883  0,4118151
##   15  0,6026742  0,4161983
##   16  0,5977382  0,4114154
##   17  0,6026742  0,4163846
##   18  0,5977096  0,4096108
##   19  0,5750487  0,3750827
##   20  0,5846430  0,3874747
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
plot(fit)
```



c. All possible predictors

```
set.seed(42)
pred_knn <- knn(train = Glass[-test, c("RI","Na","Mg" ,"Al","Si", "K", "Ca","Ba","Fe")],
                test = Glass[test, c("RI","Na","Mg" ,"Al","Si", "K", "Ca","Ba","Fe")],

                cl = Glass[-test, "Type"],
                k = 1)

table(pred_knn, Real = Glass[test, "Type"])
```

```
##          Real
## pred_knn  1  2  3  5  6  7
##        1 13  4  2  0  0  0
##        2  1 14  2  0  1  0
##        3  4  0  1  0  0  1
##        5  0  1  0  3  0  0
##        6  0  0  0  0  1  1
##        7  0  0  0  1  1  6
```

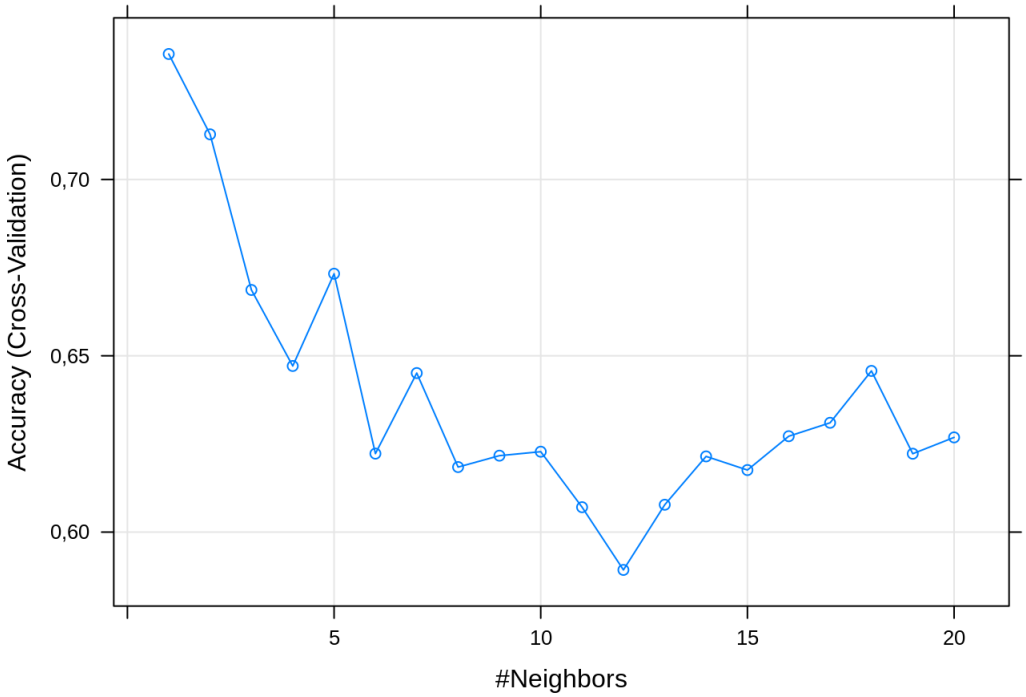
```
mean(pred_knn == Glass[test, "Type"])
```

```
## [1] 0,6666667
```

```
set.seed(42)

trControl <- trainControl(method  = "cv",
                          number  = 12)

fit <- train(Type ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe,
             method     = "knn",
             tuneGrid   = expand.grid(k = 1:20),
             trControl  = trControl,
             metric     = "Accuracy",
             data       = Glass)
fit
```

```
## k-Nearest Neighbors
##
## 214 samples
##   9 predictor
##   6 classes: '1', '2', '3', '5', '6', '7'
##
## No pre-processing
## Resampling: Cross-Validated (12 fold)
## Summary of sample sizes: 198, 197, 197, 195, 196, 197, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0,7355790  0,6420605
##    2  0,7127928  0,6096557
##    3  0,6686697  0,5452538
##    4  0,6471090  0,5079293
##    5  0,6732707  0,5469701
##    6  0,6222588  0,4758995
##    7  0,6451005  0,5044083
##    8  0,6184461  0,4674469
##    9  0,6216801  0,4692021
##   10  0,6228034  0,4705830
##   11  0,6070656  0,4486693
##   12  0,5892780  0,4252646
##   13  0,6077625  0,4479702
##   14  0,6214704  0,4634332
##   15  0,6176005  0,4598108
##   16  0,6271948  0,4715813
##   17  0,6310020  0,4759820
##   18  0,6457133  0,4969750
##   19  0,6222301  0,4598929
##   20  0,6268597  0,4668207
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

```
plot(fit)
```



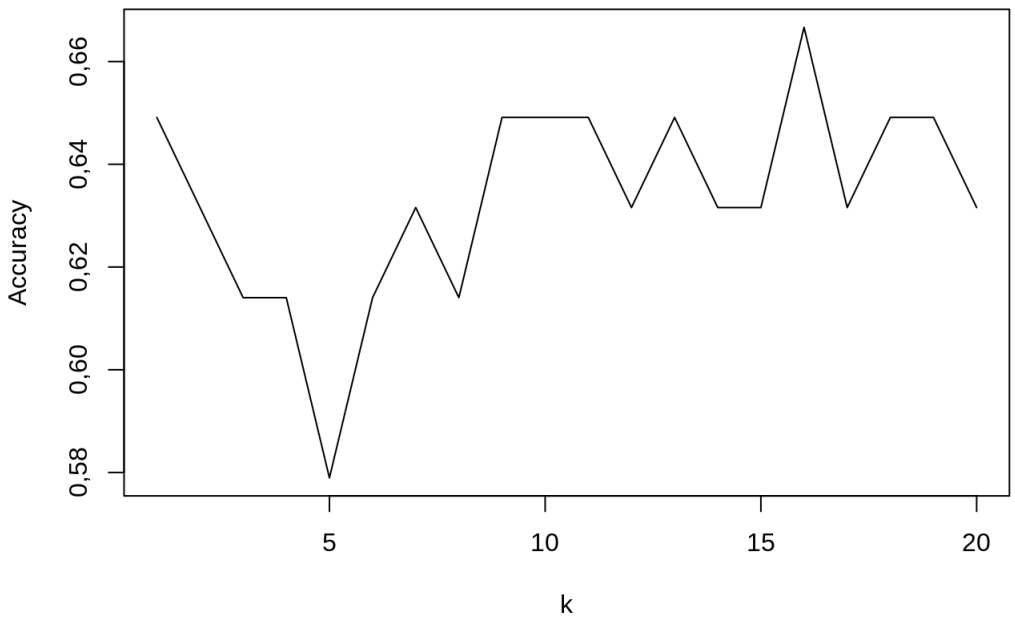
d. with PCA

```
pca.result <- prcomp(Glass[-test,-c(10:11)], scale=T)
train.pca <- pca.result$x
test.pca <- predict(pca.result, Glass[test,-c(10:11)])

k <- NULL
accuracy <- NULL
set.seed(42)
for (i in 1:20){
  pred_knn <- knn(train.pca, test.pca, Glass[-test, "Type"], k=i)
  accuracy <- c(accuracy, mean(pred_knn == Glass[test, "Type"]))
  k <- c(k, i)
}
result <- data.frame('k' = k, 'Accuracy' = accuracy)
```

```
result
```

```
##     k  Accuracy
## 1   1 0,6491228
## 2   2 0,6315789
## 3   3 0,6140351
## 4   4 0,6140351
## 5   5 0,5789474
## 6   6 0,6140351
## 7   7 0,6315789
## 8   8 0,6140351
## 9   9 0,6491228
## 10 10 0,6491228
## 11 11 0,6491228
## 12 12 0,6315789
## 13 13 0,6491228
## 14 14 0,6315789
## 15 15 0,6315789
## 16 16 0,6666667
## 17 17 0,6315789
## 18 18 0,6491228
## 19 19 0,6491228
## 20 20 0,6315789
```

```
plot(result, type = 'l')
```




```
result[which.max(result$Accuracy),]
```

```
##    k  Accuracy
## 16 16 0,6666667
```

e. Comparing all the models

```
##            Model  Accuracy
## 1             Al 0,5919404
## 2   Ri + Al + Si 0,6233678
## 3 All predictors 0,7400707
## 4            PCA 0,6842105
```

3. Logistic regression

“Ri”

```
glm_fit_ri <- glm(Type ~ Ri, data = Glass,
                  family = "binomial")
summary(glm_fit_ri)
```

```
##
## Call:
## glm(formula = Type ~ Ri, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,6522  -1,4544   0,8472   0,8865   1,2360
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    84,85      71,45   1,188    0,235
## Ri            -55,40      47,05  -1,178    0,239
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 269,16  on 212  degrees of freedom
## AIC: 273,16
##
## Number of Fisher Scoring iterations: 4
```

“Na”

```
glm_fit_na <- glm(Type ~ Na, data = Glass,
                  family = "binomial")
summary(glm_fit_na)
```

```
##
## Call:
## glm(formula = Type ~ Na, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,7436  -1,3976   0,7987   0,9139   1,3125
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4,5226     2,5630  -1,765   0,0776 .
## Na            0,3924     0,1921   2,043   0,0411 *
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 266,10  on 212  degrees of freedom
## AIC: 270,1
##
## Number of Fisher Scoring iterations: 4
```

“Mg”

```
glm_fit_mg <- glm(Type ~ Mg, data = Glass,
                  family = "binomial")
summary(glm_fit_mg)
```

```
##
## Call:
## glm(formula = Type ~ Mg, family = "binomial", data = Glass)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1,85959  -1,17969   0,07464   1,05855   1,39758
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   5,8820     1,4769   3,983 6,81e-05 ***
## Mg           -1,6045     0,4201  -3,820 0,000134 ***
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 214,97  on 212  degrees of freedom
## AIC: 218,97
##
## Number of Fisher Scoring iterations: 7
```

“Al”

```
glm_fit_al <- glm(Type ~ Al, data = Glass,
                  family = "binomial")
summary(glm_fit_al)
```

```
##
## Call:
## glm(formula = Type ~ Al, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,8816  -1,1959   0,5304   0,8195   2,0056
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2,7368     0,6500  -4,210 2,55e-05 ***
## Al            2,5563     0,4857   5,263 1,41e-07 ***
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 228,87  on 212  degrees of freedom
## AIC: 232,87
##
## Number of Fisher Scoring iterations: 5
```

“Si”

```
glm_fit_si <- glm(Type ~ Si, data = Glass,
                  family = "binomial")
summary(glm_fit_si)
```

```
##
## Call:
## glm(formula = Type ~ Si, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,5324  -1,4729   0,8768   0,8911   0,9743
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4,99094   13,62203  -0,366    0,714
## Si            0,07863    0,18753   0,419    0,675
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 270,37  on 212  degrees of freedom
## AIC: 274,37
##
## Number of Fisher Scoring iterations: 4
```

“K”

```
glm_fit_k <- glm(Type ~ K, data = Glass,
                  family = "binomial")
summary(glm_fit_k)
```

```
##
## Call:
## glm(formula = Type ~ K, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,5153  -1,4669   0,8791   0,8999   0,9283
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0,6187     0,1960   3,157   0,0016 **
## K             0,2142     0,2838   0,755   0,4504
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 269,86  on 212  degrees of freedom
## AIC: 273,86
##
## Number of Fisher Scoring iterations: 4
```

“Ca”

```
glm_fit_ca <- glm(Type ~ Ca, data = Glass,
                  family = "binomial")
summary(glm_fit_ca)
```

```
##
## Call:
## glm(formula = Type ~ Ca, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,5675  -1,4670   0,8724   0,9152   1,0621
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0,4157     1,0055  -0,413    0,679
## Ca            0,1276     0,1123   1,137    0,256
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 269,16  on 212  degrees of freedom
## AIC: 273,16
##
## Number of Fisher Scoring iterations: 4
```

“Ba”

```
glm_fit_ba <- glm(Type ~ Ba, data = Glass,
                  family = "binomial")
summary(glm_fit_ba)
```

```
##
## Call:
## glm(formula = Type ~ Ba, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2,4432  -1,3938   0,9755   0,9755   0,9755
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0,4955     0,1532   3,234  0,00122 **
## Ba            3,5323     1,5720   2,247  0,02464 *
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 250,79  on 212  degrees of freedom
## AIC: 254,79
##
## Number of Fisher Scoring iterations: 7
```

“Fe”

```
glm_fit_fe <- glm(Type ~ Fe, data = Glass,
                  family = "binomial")
summary(glm_fit_fe)
```

```
##
## Call:
## glm(formula = Type ~ Fe, family = "binomial", data = Glass)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1,4951  -1,4949   0,8901   0,8902   0,8902
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0,72123    0,16891   4,270 1,96e-05 ***
## Fe           0,00147    1,49893   0,001    0,999
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 270,54  on 212  degrees of freedom
## AIC: 274,54
##
## Number of Fisher Scoring iterations: 4
```

“All predictors”

```
glm_fit_all <- glm(Type ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe , data = Glass,
                  family = "binomial")
summary(glm_fit_all)
```

```
##
## Call:
## glm(formula = Type ~ RI + Na + Mg + Al + Si + K + Ca + Ba + Fe,
##     family = "binomial", data = Glass)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1,82235  -0,75017   0,04975   0,72806   2,19096
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 575,3570   341,4893   1,685  0,09202 .
## RI          -67,1196   192,6359  -0,348  0,72752
## Na           -3,5201     2,0937  -1,681  0,09271 .
## Mg           -6,1380     2,2075  -2,780  0,00543 **
## Al           -1,4019     2,3365  -0,600  0,54850
## Si           -5,0003     2,0758  -2,409  0,01600 *
## K            -4,4712     2,4649  -1,814  0,06969 .
## Ca           -4,3862     2,1879  -2,005  0,04499 *
## Ba           -4,9974     2,5373  -1,970  0,04889 *
## Fe            0,9779     2,0595   0,475  0,63489
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 178,67  on 204  degrees of freedom
## AIC: 198,67
##
## Number of Fisher Scoring iterations: 7
```

“Best”

```
glm_fit_best <- glm(Type ~ Mg + Al + Si , data = Glass,
                    family = "binomial")
summary(glm_fit_best)
```

```
##
## Call:
## glm(formula = Type ~ Mg + Al + Si, family = "binomial", data = Glass)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1,79584  -0,88280   0,06546   0,83801   2,15389
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  56,5629    26,6852   2,120 0,034036 *
## Mg           -1,5989     0,4144  -3,858 0,000114 ***
## Al            3,1030     0,6993   4,437 9,11e-06 ***
## Si           -0,7529     0,3602  -2,090 0,036610 *
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 189,31  on 210  degrees of freedom
## AIC: 197,31
##
## Number of Fisher Scoring iterations: 7
```

Transfromation

```
one <- Glass$Mg
two <-  Glass$Al^2
three <- Glass$Si^3

glm_fit_transf <- glm(Type ~ one + two + three , data = Glass,
                      family = "binomial")
summary(glm_fit_transf)
```

```
##
## Call:
## glm(formula = Type ~ one + two + three, family = "binomial",
##     data = Glass)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1,95507  -0,84954   0,06482   0,77752   2,08106
##
## Coefficients:
```

```
## Warning in printCoefmat(coefs, digits = digits, signif.stars = signif.stars, : в
## результате преобразования созданы NA
```

```
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2,082e+01  9,388e+00   2,218 0,026538 *
## one         -1,507e+00  4,055e-01  -3,717 0,000202 ***
## two          1,410e+00  3,077e-01   4,581 4,63e-06 ***
## three       -4,617e-05  2,278e-05  -2,027 0,042648 *
## ---
## Signif. codes:  0 '***' 0,001 '**' 0,01 '*' 0,05 '.' 0,1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 270,54  on 213  degrees of freedom
## Residual deviance: 185,78  on 210  degrees of freedom
## AIC: 193,78
##
## Number of Fisher Scoring iterations: 7
```

```
glm_fit_transf2 <- glm(Type ~ Mg + Al + Si , data = Glass[-test,],
                      family = "binomial")
pred_glm <- predict(glm_fit_transf2, type = "response", newdata = Glass[test, ]) > .5
table(pred_glm, Real = Glass[test, 10])
```

```
##         Real
## pred_glm  1  2  3  5  6  7
##    FALSE 10  3  0  0  0  0
##    TRUE   8 16  5  4  3  8
```

```
16/20
```

```
## [1] 0,8
```