Classification_models_1

Khanin Sisaengsuwanchai

3/23/2022

The goal is to help answer whether maternal smoking has an effect on birth weight, applying LDA, QDA, NaiveBayes .

Let's start by loading the data

```
# Load the "infants" dataset
load(url("http://www.stodden.net/StatData/KaiserBabies.rda"))

# Check the data
names(infants)
```

```
##  [1] "gestation" "bwt"       "parity"    "age"       "ed"        "ht"
##  [7] "wt"        "dage"      "ded"       "dht"       "dwt"       "marital"
## [13] "inc"       "smoke"     "number"
```

```
dim(infants)
```

```
## [1] 1236   15
```

```
## Understand the data
summary(infants)
```

```
##    gestation          bwt           parity            age
##  Min.   :148.0   Min.   : 55.0   Min.   : 0.000   Min.   :15.00
##  1st Qu.:272.0   1st Qu.:108.8   1st Qu.: 0.000   1st Qu.:23.00
##  Median :280.0   Median :120.0   Median : 1.000   Median :26.00
##  Mean   :279.3   Mean   :119.6   Mean   : 1.932   Mean   :27.26
##  3rd Qu.:288.0   3rd Qu.:131.0   3rd Qu.: 3.000   3rd Qu.:31.00
##  Max.   :353.0   Max.   :176.0   Max.   :13.000   Max.   :45.00
##  NA's   :13                                       NA's   :2
##                 ed             ht              wt             dage
##  No High School   : 19   Min.   :53.00   Min.   : 87.0   Min.   :18.00
##  Some High School:183   1st Qu.:62.00   1st Qu.:114.8   1st Qu.:25.00
##  High School     :444   Median :64.00   Median :125.0   Median :29.00
##  Trade           : 65   Mean   :64.05   Mean   :128.6   Mean   :30.35
##  Some College    :298   3rd Qu.:66.00   3rd Qu.:139.0   3rd Qu.:34.00
##  College         :219   Max.   :72.00   Max.   :250.0   Max.   :62.00
##  Unknown          :  8   NA's   :22      NA's   :36      NA's   :7
##                ded           dht            dwt           marital
##  No High School  : 33   Min.   :60.0   Min.   :110.0   Married:1208
##  Some High School:193   1st Qu.:68.0   1st Qu.:155.0   Once   :  20
##  High School     :342   Median :71.0   Median :170.0   Never  :   6
##  Trade           : 37   Mean   :70.2   Mean   :171.2   NA's   :   2
##  Some College    :265   3rd Qu.:72.0   3rd Qu.:185.0
##  College         :347   Max.   :78.0   Max.   :260.0
##  Unknown         : 19   NA's   :492    NA's   :499
##              inc                  smoke         number
##  [2500, 5000)  :195   Never          :544   Never  :544
##  [6000, 7000)  :180   Now            :484   20-29  :195
##  [5000, 6000)  :179   Until Pregnant: 95   5-9    :167
##  [10000, 12500):143   Once, Not Now :103   1-4    :155
##  [7000, 8000)  :138   Unknown       : 10   10-14  : 75
##  [8000, 9000)  :126                        30-39  : 32
##  (Other)       :275                        (Other): 68
```

According to the objective to examine the effect of maternal smoking on birthweight, I would instead use birth weight to understand the effect and predict maternal smoking because LDA, QDA, and NaiveBayes are classifiers not regression problem.

Data preprocessing

```
## Since dht and dwt have a lot of nulls,  fill missing values for dht and dwt with mean of its value
infants$dht[is.na(infants$dht)] <- mean(infants$dht, na.rm = T)
infants$dwt[is.na(infants$dwt)] <- mean(infants$dwt, na.rm = T)
infants$wt[is.na(infants$wt)] <- mean(infants$wt, na.rm = T)

# Remove null from the data
infants = infants[complete.cases(infants), ]

# Recheck the data manipulation
summary(infants)
```

```
##    gestation          bwt            parity            age
##  Min.   :181.0   Min.   : 55.0   Min.   : 0.000   Min.   :15.00
##  1st Qu.:272.0   1st Qu.:108.0   1st Qu.: 0.000   1st Qu.:23.00
##  Median :280.0   Median :120.0   Median : 1.000   Median :26.00
##  Mean   :279.3   Mean   :119.6   Mean   : 1.905   Mean   :27.22
##  3rd Qu.:288.0   3rd Qu.:131.0   3rd Qu.: 3.000   3rd Qu.:31.00
##  Max.   :353.0   Max.   :176.0   Max.   :11.000   Max.   :45.00
##
##                 ed            ht              wt             dage
##  No High School  : 19   Min.   :53.00   Min.   : 87.0   Min.   :18.00
##  Some High School:174   1st Qu.:62.00   1st Qu.:115.0   1st Qu.:25.00
##  High School     :431   Median :64.00   Median :125.0   Median :29.00
##  Trade           : 63   Mean   :64.04   Mean   :128.6   Mean   :30.36
##  Some College    :285   3rd Qu.:66.00   3rd Qu.:138.2   3rd Qu.:34.00
##  College         :213   Max.   :72.00   Max.   :250.0   Max.   :62.00
##  Unknown         :  7
##                  ded           dht             dwt           marital
##  No High School  : 33   Min.   :60.00   Min.   :110.0   Married:1170
##  Some High School:183   1st Qu.:70.00   1st Qu.:165.0   Once   :  17
##  High School     :329   Median :70.20   Median :171.2   Never  :   5
##  Trade           : 37   Mean   :70.23   Mean   :171.2
##  Some College    :257   3rd Qu.:71.00   3rd Qu.:175.0
##  College         :336   Max.   :78.00   Max.   :260.0
##  Unknown         : 17
##                inc                  smoke         number
##  [2500, 5000)  :188   Never         :526   Never  :526
##  [6000, 7000)  :173   Now           :465   20-29  :187
##  [5000, 6000)  :171   Until Pregnant: 92   5-9    :162
##  [10000, 12500):138   Once, Not Now : 99   1-4    :149
##  [7000, 8000)  :133   Unknown       : 10   10-14  : 71
##  Unknown       :123                        30-39  : 30
##  (Other)       :266                        (Other): 67
```

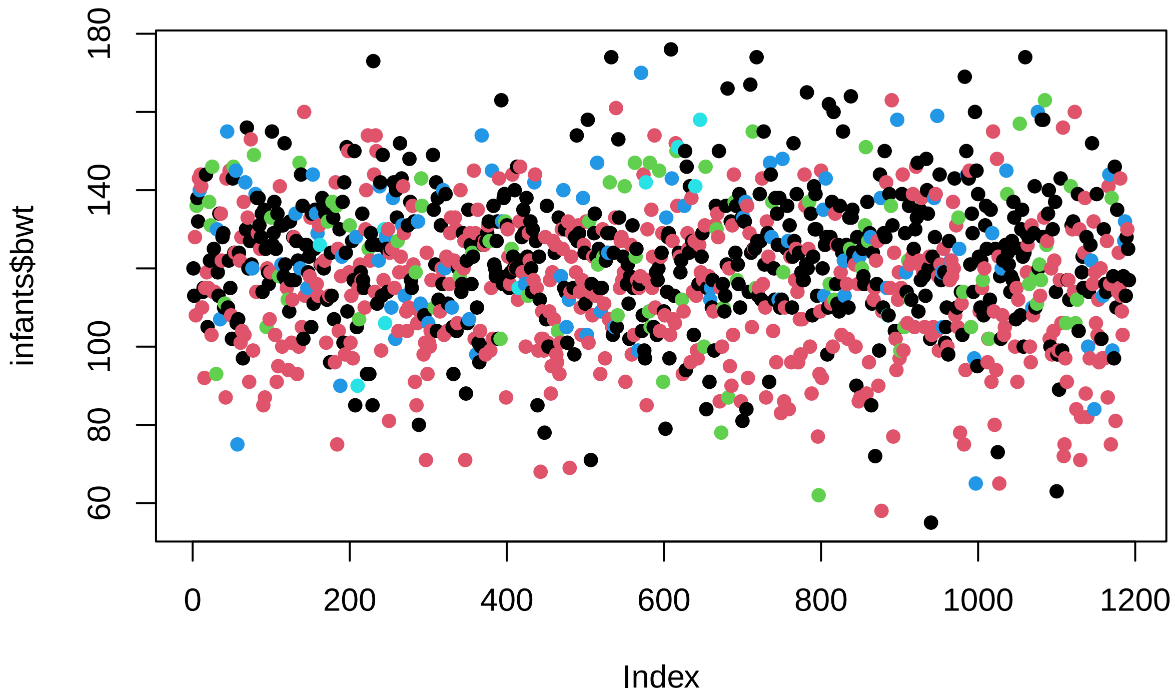
```
dim(infants) # The rows go down from 1236 to 1192
```

```
## [1] 1192   15
```

Now all the NAs are gone

Data understanding

```
# A scatter plot that shows the points in groups according to their "maternal smoke"
plot(infants$bwt,
pch = 16,
col = as.numeric(infants$smoke)
)
```



Based on the plot, there is no clear patterns between maternal smoke and birthweight.

Therefore, we need to investigate the relationships using other statistical methods and models.

```
# Correlation plot with qualitative data removal
coorelation = cor(infants[,  -c(5, 9, 12, 13, 14, 15)])
coorelation
```

```
##              gestation        bwt      parity           age           ht
## gestation  1.000000000 0.42279626 -0.10584995 -0.0569947688  0.072589084
## bwt        0.422796257 1.00000000  0.02522582  0.0290861580  0.201689082
## parity    -0.105849955 0.02522582  1.00000000  0.5307818279 -0.028562995
## age       -0.056994769 0.02908616  0.53078183  1.0000000000 -0.006682069
## ht         0.072589084 0.20168908 -0.02856300 -0.0066820692  1.000000000
## wt         0.027338316 0.15805223  0.18862310  0.1484914054  0.430845238
## dage      -0.032252953 0.03914802  0.47678154  0.8199498962 -0.029637908
## dht        0.003283079 0.08231169 -0.05045843 -0.0457083338  0.268630217
## dwt        0.003303946 0.11110868  0.05956771 -0.0002038911  0.198270487
##                   wt        dage          dht           dwt
## gestation 0.02733832 -0.03225295  0.003283079  0.0033039456
## bwt       0.15805223  0.03914802  0.082311686  0.1111086765
## parity    0.18862310  0.47678154 -0.050458427  0.0595677133
## age       0.14849141  0.81994990 -0.045708334 -0.0002038911
## ht        0.43084524 -0.02963791  0.268630217  0.1982704870
## wt        1.00000000  0.18205187  0.105631547  0.1512449825
## dage      0.18205187  1.00000000 -0.114784637 -0.0245753234
## dht       0.10563155 -0.11478464  1.000000000  0.5351696900
## dwt       0.15124498 -0.02457532  0.535169690  1.0000000000
```

According to the correlation of the quantitative data, age and dage with a correlation value of 0.82 and dht and dwt with a correlation value of 0.54 have significant correlation values, therefore I will choose only age and dht as representatives because I can randomly pick one from the group.

Initially, we need to split the data into 10-folds cross validation since this k neither suffers from excessively high bias nor very high variance compared to LOOCV.

```
library(caret) # I'll use the caret package to slice the dataset into 10 folds
```

```
## Loading required package: lattice
```

```
set.seed(1)
infants_folds = createFolds(infants$smoke, k = 10)
# From the data of 1,192 rows, I will split the data into 10 training and test dataset
# when k = 10. Each training dataset has 1111 datapoints, and test dataset has 120 datapoints.

infants_folds[1:3] # Note that the values of this list is index, not the actual values.
```

```
## $Fold01
##   [1]    1   10   28   40   59   62   81   83  106  111  123  146  147  152  196
##  [16]  202  208  229  248  263  277  300  310  321  328  341  350  355  375  407
##  [31]  415  418  424  431  440  442  454  468  478  479  484  490  502  510  532
##  [46]  544  561  573  579  583  585  590  599  600  609  622  630  643  645  652
##  [61]  656  662  672  693  702  728  736  754  762  763  767  771  779  784  802
##  [76]  803  804  810  812  819  822  828  834  836  837  839  846  850  869  876
##  [91]  877  905  930  933  938  944  953  967  970  982  998 1012 1014 1024 1031
## [106] 1037 1045 1054 1065 1067 1073 1085 1089 1092 1095 1097 1112 1129 1172 1178
##
## $Fold02
##   [1]   32   38   39   46   60   68   77   95   97  101  107  148  158  168  169
##  [16]  181  193  204  207  209  211  241  244  246  253  269  270  280  286  291
##  [31]  301  302  303  312  320  340  343  345  390  391  393  395  396  398  399
##  [46]  401  411  417  420  425  428  444  445  475  488  508  511  515  524  538
##  [61]  539  546  553  556  565  614  619  651  655  659  667  680  687  690  716
##  [76]  718  721  731  756  760  764  793  809  811  816  849  860  865  873  874
##  [91]  895  901  910  931  936  955  964  971  985  992 1013 1018 1041 1100 1105
## [106] 1117 1136 1141 1144 1145 1151 1157 1160 1161 1168 1179 1182 1185 1186
##
## $Fold03
##   [1]   21   25   56   65   67  104  119  125  127  142  150  157  159  163  170
##  [16]  175  180  185  186  189  213  214  224  240  242  245  247  249  257  262
##  [31]  274  276  288  293  344  346  361  374  380  404  430  458  467  480  481
##  [46]  487  492  496  517  520  521  531  549  552  555  558  563  574  578  581
##  [61]  598  626  637  642  648  654  668  682  713  715  726  733  734  739  740
##  [76]  742  744  778  782  801  821  843  875  887  914  927  929  934  946  976
##  [91]  980 1001 1002 1004 1007 1020 1021 1038 1040 1048 1052 1056 1058 1066 1098
## [106] 1113 1114 1119 1120 1127 1162 1167 1171 1173 1175 1188 1189 1190
```

1. LDA

Implement a function for LDA that uses the K-fold dataset.

```
library(MASS)
## Create a function that takes the index of each fold, fit and predict using LDA
cV.LDA = function(complete_data, i){

  # Training data: not in fold indices
  train = complete_data[-i, ]

  # Test data: all data in the fold indices
  test = complete_data[i, ]

  # fit all variables except dage and dwt due to high correlation
  # ded and ed are very similar so I pick one of them
  lda.fit = lda(smoke ~ . - dage - dwt - ded, data = train)

  # apply the model to the test dataset and obtain predicted classes
  lda.class = predict(lda.fit, test)$class

  # return a data.frame with two columns containing the cross-validated
  # predictions for the fold and the corrsponding reference observations
  return(data.frame(Prediction = lda.class,
                    Actual = test$smoke)) # Obtain actual class from the test dataset
}
```

```
# Check test errors for only one class
cV_one_fold = cV.LDA(infants, infants_folds[[1]])
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

```
# Show a confusion matrix and compute test errors
table(cV_one_fold$Prediction, cV_one_fold$Actual)
```

```
##
##                    Never Now Until Pregnant Once, Not Now Unknown
##   Never               53   0              0             0       0
##   Now                  0  40              6             6       0
##   Until Pregnant       0   4              2             1       0
##   Once, Not Now        0   3              1             2       0
##   Unknown              0   0              0             1       1
```

```
mean(cV_one_fold$Prediction != cV_one_fold$Actual)  # Error rate
```

```
## [1] 0.1833333
```

Calculate test errors for all cross validation dataset with k = 10

```
# create empty data.frame
lda.cV_all_folds = data.frame(Prediction = numeric(0), Reference = numeric(0))

for(i in infants_folds){
  # add the rows of the fold
  lda.cV_all_folds = rbind(lda.cV_all_folds, cV.LDA(infants, i))
}
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear

## Warning in lda.default(x, grouping, ...): variables are collinear
```

```
# Show a confusion matrix and compute test errors
table(lda.cV_all_folds$Prediction, lda.cV_all_folds$Actual)
```

```
##
##                   Never Now Until Pregnant Once, Not Now Unknown
##    Never            526   5              3             5       0
##    Now                0 397             60            54       0
##    Until Pregnant     0  43             20            19       0
##    Once, Not Now      0  17              8            15       0
##    Unknown            0   3              1             6      10
```

```
mean(lda.cV_all_folds$Prediction != lda.cV_all_folds$Actual)  # Error rate
```

```
## [1] 0.1879195
```

2. QDA

Implement a function for QDA that uses the K-fold dataset.

```
## Create a function that takes the index of each fold, fit and predict using QDA
cV.QDA = function(complete_data, i){

  # Training data: not in fold indices
  train = complete_data[-i, ]

  # Test data: all data in the fold indices
  test = complete_data[i, ]

  # fit a model using the training dataset
  # I choose to remove all qualitative variables because the QDA model gives an error
  qda.fit = qda(smoke ~ . - dage - dwt -ed -ded -marital -inc -number, data = train)

  # apply the model to the test dataset and obtain predicted classes
  qda.class = predict(qda.fit, test)$class

  # return a data.frame with two columns containing the cross-validated
  # predictions for the fold and the corrsponding reference observations
  return(data.frame(Prediction = qda.class,
                    Actual = test$smoke)) # Obtain actual class from the test dataset
}

# Call the above function, train, and predict using QDA

# create empty data.frame
qda.cV_all_folds = data.frame(Prediction = numeric(0), Reference = numeric(0))

for(i in infants_folds){
  # add the rows of the fold
  qda.cV_all_folds = rbind(qda.cV_all_folds, cV.QDA(infants, i))
}

# Show a confusion matrix and compute test errors
table(qda.cV_all_folds$Prediction, qda.cV_all_folds$Actual)
```

```
##
##                    Never Now Until Pregnant Once, Not Now Unknown
##   Never            344 213             61            63       8
##   Now              174 248             31            34       2
##   Until Pregnant     2   1              0             0       0
##   Once, Not Now      4   3              0             1       0
##   Unknown            2   0              0             1       0
```

```
mean(qda.cV_all_folds$Prediction != qda.cV_all_folds$Actual)  # Error rate
```

```
## [1] 0.5025168
```

3.

Implement a function for NaiveBayes that uses the K-fold dataset.

```
library(e1071)
## Create a function that takes the index of each fold, fit and predict using NaiveBayes

cV.NB = function(complete_data, i){
  # Training data: not in fold indices
  train = complete_data[-i, ]

  # Test data: all data in the fold indices
  test = complete_data[i, ]

  # fit all variables except dage and dwt due to high correlation
  # ded and ed are very similar so I pick one of them
  nb.fit = naiveBayes(smoke ~ . - dage - dwt -ded, data = train)

  # apply the model to the test dataset and obtain predicted classes
  nb.class = predict(nb.fit, test)

  # return a data.frame with two columns containing the cross-validated
  # predictions for the fold and the corrsponding reference observations
  return(data.frame(Prediction = nb.class,
                    Actual = test$smoke)) # Obtain actual class from the test dataset
}

# Call the above function to train and predict using NaiveBayes

# create empty data.frame
nb.cV_all_folds = data.frame(Prediction = numeric(0), Reference = numeric(0))

for(i in infants_folds){
  # add the rows of the fold
  nb.cV_all_folds = rbind(nb.cV_all_folds, cV.NB(infants, i))
}

# Show a confusion matrix and compute test errors
table(nb.cV_all_folds$Prediction, nb.cV_all_folds$Actual)
```

```
##
##                   Never Now Until Pregnant Once, Not Now Unknown
##   Never            526   0              0             0       0
##   Now                0 426             75            70       1
##   Until Pregnant     0  16              9             6       0
##   Once, Not Now      0  22              7            22       4
##   Unknown            0   1              1             1       5
```

```
mean(nb.cV_all_folds$Prediction != nb.cV_all_folds$Actual)  # Error rate
```

```
## [1] 0.1711409
```

Interpretation

```
# These are the test errors of the methods.
#         Test errors
# 1. LDA  0.1879195
# Interpretation: the overall accuracy of this model is about 81%, which is quite decent.
# Therefore, according to this model, birthweight together with additional factors
# perform well in predicting and explaining maternal smoking.
# Similarly, we can infer that maternal smoking has an impact on birthweight because of
# a reverse effect.

#         Test errors
# 2. QDA  0.5025168
# Interpretation: the overall accuracy of this model is about 50%, which is closed to
# random chance. According to this model, birthweight together with additional factors do not have
# significant effects on maternal smoking, and therefore maternal smoking does not have
# an impact on birthweight with non-linear relationship.

#         Test errors
# 3. NB   0.1711409
# Interpretation: the overall accuracy of this model is about 83%, which is slightly better
# Therefore, according to this model, birthweight together with additional factors
# perform well in predicting and explaining maternal smoking.
# Similarly, we can infer that maternal smoking has an impact on birthweight because of
# a reverse effect.
```

To compare the performance of these models, I would use test errors as a measurement.
NaiveBayes is the most accurate model of these three because it has the lowest test errors of 0.1711409, following by LDA which has test errors of 0.1879195 and QDA that performs poorly with test errors of 0.5025168.
What make differences between these models are the assumptions that each model holds. To be more specific, LDA assumes that each class from 1…K has a common covariance and that the observations are drawn from a multivariate Gaussian distribution, leading to potentially high bias and low variance trade-off. Even though QDA assumes that the observations are drawn from a multivariate Gaussian distribution, unlike LDA, it does not each class has its own covariance. Finally, NaiveBayes only assumes that within the kth class, the p predictors are independent. As far as I'm concerned, these methods work well when their assumptions hold true, which is, in this case, each class seems to has its own covariance and the predictors are independent.

However, I am of the opinion that the performance of LDA and NaiveBayes models is quite decent, except only the QDA method. As of now, I include only numerical data into the model because it gives errors "some group is too small for 'qda' ". Therefore, I am going to improve only qda by using dummy variables to represent qualitative predictors. Hopefully, it will solve the error.

I am going to use 'fastDummies' to handle the work.

```
library('fastDummies')

infants.tranf <- dummy_cols(infants, select_columns = c('ed', 'marital', 'inc' , 'number'),
           remove_selected_columns = TRUE)

head(infants.tranf)
```

```
##   gestation bwt parity age ht  wt dage              ded     dht      dwt
## 1       284 120      1  27 62 100   31          College 65.0000 110.0000
## 2       282 113      2  33 64 135   38          College 70.0000 148.0000
## 3       279 128      1  28 64 115   32 Some High School 70.2043 171.2008
## 4       282 108      1  23 67 125   24          College 70.2043 171.2008
## 5       286 136      4  25 62  93   28      High School 64.0000 130.0000
## 6       244 138      4  33 62 178   37     Some College 70.2043 171.2008
##            smoke ed_No High School ed_Some High School ed_High School ed_Trade
## 1          Never                 0                   0              0        0
## 2          Never                 0                   0              0        0
## 3            Now                 0                   0              1        0
## 4            Now                 0                   0              0        0
## 5 Until Pregnant                 0                   0              1        0
## 6          Never                 0                   0              1        0
##   ed_Some College ed_College ed_Unknown marital_Married marital_Once
## 1               0          1          0               1            0
## 2               0          1          0               1            0
## 3               0          0          0               1            0
## 4               0          1          0               1            0
## 5               0          0          0               1            0
## 6               0          0          0               1            0
##   marital_Never inc_< 2500 inc_[2500, 5000) inc_[5000, 6000) inc_[6000, 7000)
## 1             0          0                1                0                0
## 2             0          0                0                0                0
## 3             0          0                0                1                0
## 4             0          0                1                0                0
## 5             0          0                0                0                0
## 6             0          0                0                0                0
##   inc_[7000, 8000) inc_[8000, 9000) inc_[9000, 10000) inc_[10000, 12500)
## 1                0                0                 0                  0
## 2                1                0                 0                  0
## 3                0                0                 0                  0
## 4                0                0                 0                  0
## 5                1                0                 0                  0
## 6                0                0                 0                  0
##   inc_[12500, 15000) inc_15000+ inc_Unknown number_Never number_1-4 number_5-9
## 1                  0          0           0            1          0          0
## 2                  0          0           0            1          0          0
## 3                  0          0           0            0          1          0
## 4                  0          0           0            0          0          0
## 5                  0          0           0            0          0          1
## 6                  0          0           1            1          0          0
##   number_10-14 number_15-19 number_20-29 number_30-39 number_40-60 number_60+
## 1            0            0            0            0            0          0
## 2            0            0            0            0            0          0
## 3            0            0            0            0            0          0
## 4            0            0            1            0            0          0
## 5            0            0            0            0            0          0
## 6            0            0            0            0            0          0
##   number_Unknown
## 1              0
## 2              0
## 3              0
## 4              0
## 5              0
## 6              0
```

Redefine variables in QDA with the dummy variables

```r
## Create a function that takes the index of each fold, fit and predict using QDA
cV.QDA = function(complete_data, i){

  # Training data: not in fold indices
  train = complete_data[-i, ]

  # Test data: all data in the fold indices
  test = complete_data[i, ]

  # fit a model using the training dataset
  # I choose to remove all qualitative variables expect smoke because the QDA model gives an error
  qda.fit = qda(smoke ~ . - dage - dwt -ded, data = train)

  # apply the model to the test dataset and obtain predicted classes
  qda.class = predict(qda.fit, test)$class

  # return a data.frame with two columns containing the cross-validated
  # predictions for the fold and the corrsponding reference observations
  return(data.frame(Prediction = qda.class,
                    Actual = test$smoke)) # Obtain actual class from the test dataset
}

# Call the above function, train, and predict using QDA

# create empty data.frame
#qda.cV_all_folds = data.frame(Prediction = numeric(0), Reference = numeric(0))

#for(i in infants_folds){
  # add the rows of the fold
  #qda.cV_all_folds = rbind(qda.cV_all_folds, cV.QDA(infants.tranf, i))
#}

# Show a confusion matrix and compute test errors
#table(qda.cV_all_folds$Prediction, qda.cV_all_folds$Actual)
#mean(qda.cV_all_folds$Prediction != qda.cV_all_folds$Actual)  # Error rate



## I still get this error so I need to comment the code above.
# Error in qda.default(x, grouping, ...) :
# some group is too small for 'qda'
```

Unfortunately, I still get the same errors, meaning the dummy variables did not help solve the error. Therefore, I would recommend to use other methods instead of qda to fit the data in order to improve the model performance.