Statistical Learning, Homework #1

Simone Bellavia

2022-04-04

Introduction to analysis

The data come from a study conducted at a UK hospital, investigating the possible factors affecting the decision of pregnant women to breastfeed their babies, in order to *target breastfeeding promotions towards women with a lower probability of choosing it.*

For the study, 135 expectant mothers were asked what kind of feeding method they would use for their coming baby.

The responses were classified into two categories (variable *breast* in the dataset): the first category (coded 1) includes the cases "breastfeeding", "try to breastfeed" and "mixed breast and bottle-feeding", while the second category (coded 0) corresponds to "exclusive bottle-feeding". The possible factors, that are available in the data, are the advancement of the pregnancy (*pregnancy*), how the mothers were fed as babies (*howfed*), how the mother's friend fed their babies (*howfedfr*), if they have a partner (*partner*), their age (*age*), the age at which they left full-time education (*educat*), their ethnic group (*ethnic*) and if they have ever smoked (*smokebf*) or if they have stopped smoking (*smokenow*). All of the factors are two-level factors.

What will need to be done in the report are the following points:

- Data exploration
- Division of data into training and test sets
- Fit of the GLM model
- Fit of the k-NN classifier
- Fit of the Naïve Bayes classifier
- Evaluations of the performances and comparison of the results

Data Exploration

We start the report with a first phase of Data Exploration. In particular we go to check the variables that are present, the number of observations and their detail.

```
names(breastfeed)
```

```
##  [1] "breast"    "pregnancy" "howfed"    "howfedfr"  "partner"   "smokenow"
##  [7] "smokebf"   "age"       "educat"    "ethnic"
```

```
dim(breastfeed)
```

```
## [1] 139  10
```

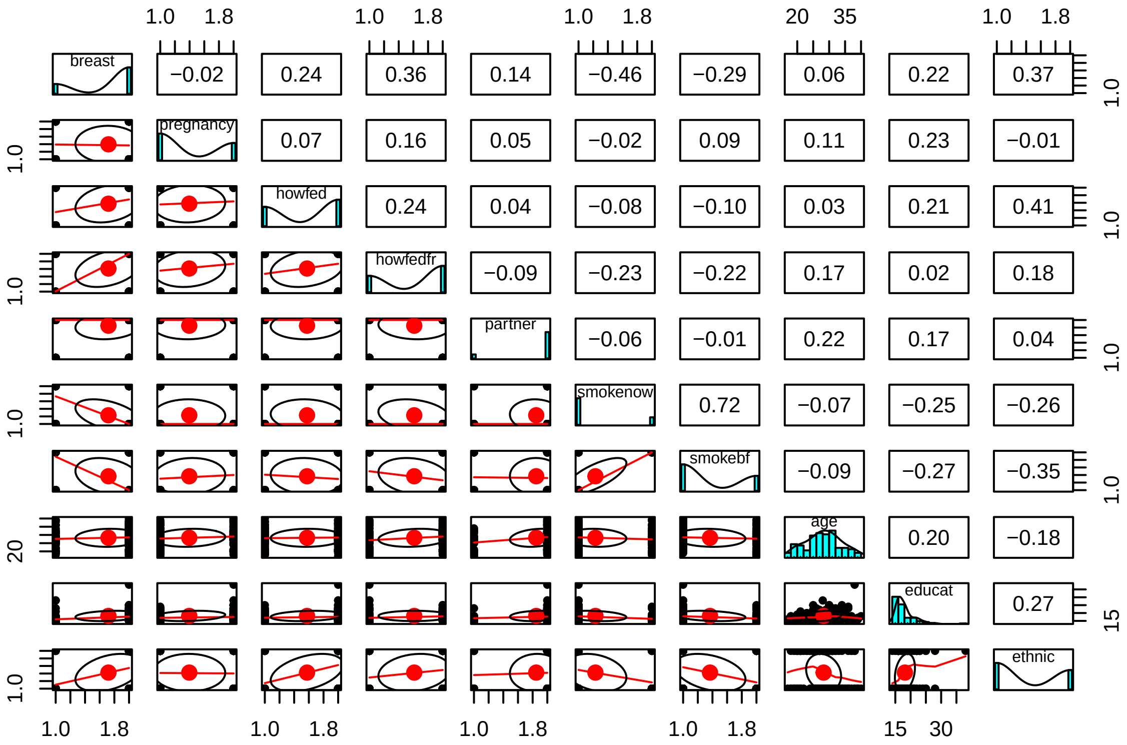
```
summary(breastfeed)
```

```
##     breast         pregnancy     howfed     howfedfr     partner    smokenow
##  Bottle: 39   End      :84   Bottle:59   Bottle:54   Single : 21   No :107
##  Breast:100   Beginning:55   Breast:80   Breast:85   Partner:118   Yes: 32
##
##
##
##
##
##  smokebf       age            educat           ethnic
##  No :88   Min.   :17.00   Min.   :14.00   White    :80
##  Yes:51   1st Qu.:25.00   1st Qu.:16.00   Non-white:59
##           Median :28.00   Median :17.00
##              Mean   :28.26   Mean   :18.15
##              3rd Qu.:32.00   3rd Qu.:19.00
##              Max.   :40.00   Max.   :38.00
##              NA's   :2       NA's   :2
```

We have 10 variables and 139 observations. From the summary of our dataset, we can see that all variables are qualitative, non-numeric and binary, except for *age* and *educat* that are continuous.

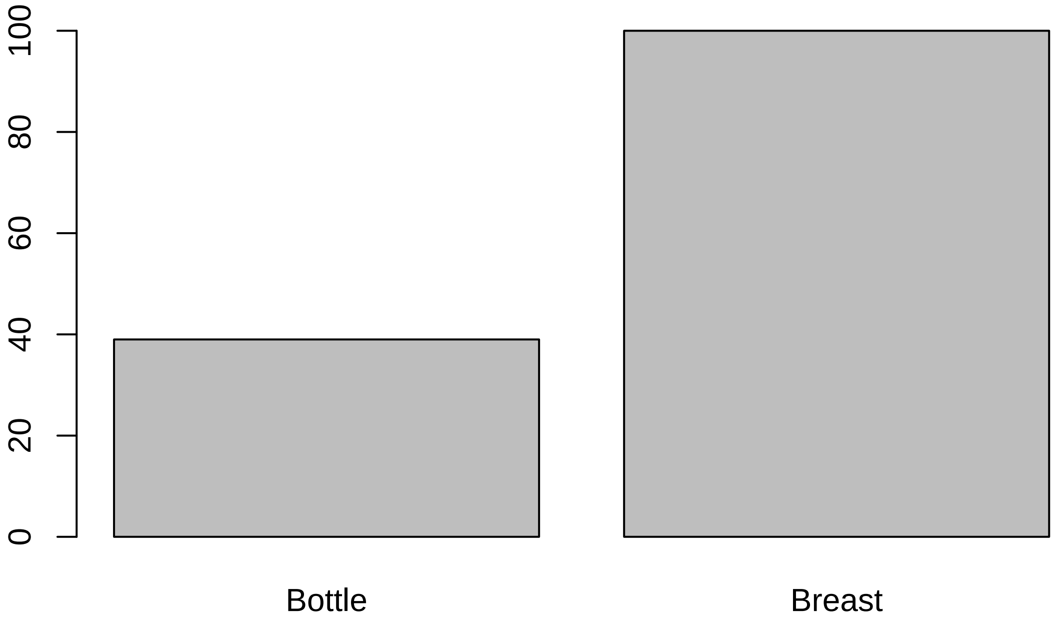
Through the scatter plot matrix it is possible to see all the different Pearson correlations within the variables and the distributions.

```
pairs.panels(breastfeed)
```



It immediately jumps out that the variable *age* assumes almost a normal distribution. All the others are binary variables that follow their own distribution. Some correlations give us an idea of how they relate to the breast variable, although it is a rough idea. We will plot an histogram to check in detail the distribution of the response variable *breast*.

```
plot(breastfeed$breast)
```



The distribution, as expected, is binary and is divided into "Bottle" and "Breast," which are two qualitative categories. At first sight, it can be seen that the majority of expectant mothers (71%) prefer a method that tends towards breastfeeding, while the remaining 29% prefer exclusive bottle-feeding.

From the summary it is also possible to detect 4 observations that contain missing values. So we can't proceed unless we pre-process these rows first, wtherwise we will not be able to use the GLM model. With *na.omit* function all incomplete cases are removed and the new dataframe is cloned into *processed_breastfeed* (which we will reuse later).

```
processed_breastfeed = na.omit(breastfeed)
```

For the purpose of GLM model fit, it is not necessary to proceed with data preprocessing. We can use the new dataframe we created without missing values. But first, we will divide the data into training and test sets.

Splitting the data into training and test sets

We will set the seed to make the object reproducible. Then, we will partition 75% of the data into training set and the remaining 25% into the test set, by generating sampled and non-sequential indexes. The new sets will be called *glm_train* and *glm_test*.

```
# setting the seed to make the partition reproducible
set.seed(99)

# 75% of the sample size
smp_size <- floor(0.75 * nrow(processed_breastfeed))

train_ind <- sample(seq_len(nrow(processed_breastfeed)), size = smp_size)

glm_train <- processed_breastfeed[train_ind, ]
glm_test <- processed_breastfeed[-train_ind, ]
```

With these two new sets, we can move forward with the fit of the statistical models.

GLM model

Fitting the GLM Model

We now want to predict the `breast` target variable using a multivariate Logistic Regression model as follows:

$logit(E(breast)) = \beta_0 + \beta_1 pregnancy + \beta_2 howfed + \beta_3 howfedfr + \beta_4 partner + \beta_5 age + \beta_6 educat + \beta_7 ethnic + \beta_8 smokenow + \beta_9 smokebf$

We will set up the model:

```
glm.fits <- glm(breast ~ pregnancy + howfed + howfedfr + partner + age
                + educat + ethnic + smokenow + smokebf,
                data=processed_breastfeed,
                family=binomial) # family=binomial selects a Logistic Regression model
glm.fits
```

```
##
## Call:  glm(formula = breast ~ pregnancy + howfed + howfedfr + partner +
##     age + educat + ethnic + smokenow + smokebf, family = binomial,
##     data = processed_breastfeed)
##
## Coefficients:
##        (Intercept)  pregnancyBeginning        howfedBreast      howfedfrBreast
##           -4.50386            -0.98115             0.30804             1.49555
##     partnerPartner                 age              educat    ethnicNon-white
##            1.08438             0.02681             0.17400             1.95507
##        smokenowYes          smokebfYes
##           -3.31232             1.74417
##
## Degrees of Freedom: 134 Total (i.e. Null);  125 Residual
## Null Deviance:       156.6
## Residual Deviance: 94.4  AIC: 114.4
```

Inspecting the GLM object

Let's inspect the GLM object to retrieve the informations that we generated, by calling the *summary* function.

```
summary(glm.fits)
```

```
##
## Call:
## glm(formula = breast ~ pregnancy + howfed + howfedfr + partner +
##     age + educat + ethnic + smokenow + smokebf, family = binomial,
##     data = processed_breastfeed)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6515  -0.4189   0.2285   0.4594   2.7506
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -4.50386    2.43357  -1.851  0.06421 .
## pregnancyBeginning -0.98115    0.57740  -1.699  0.08927 .
## howfedBreast        0.30804    0.59119   0.521  0.60233
## howfedfrBreast      1.49555    0.58784   2.544  0.01095 *
## partnerPartner      1.08438    0.70281   1.543  0.12285
## age                 0.02681    0.05151   0.520  0.60279
## educat              0.17400    0.12703   1.370  0.17077
## ethnicNon-white     1.95507    0.75601   2.586  0.00971 **
## smokenowYes        -3.31232    1.01311  -3.269  0.00108 **
## smokebfYes          1.74417    1.00626   1.733  0.08304 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 156.58  on 134  degrees of freedom
## Residual deviance:  94.40  on 125  degrees of freedom
## AIC: 114.4
##
## Number of Fisher Scoring iterations: 6
```

Apparently, women early in pregnancy are less likely to decide to breastfeed. Another factor that seems to influence the choice to breastfeed is how the mother's friends decided to breastfeed. Women with partners are more likely to decide to breastfeed. A very interesting fact is that women of non-white ethnicity tend to choose to breastfeed. Women who smoke decide to avoid breastfeeding and lean toward bottle-feeding; instead, those who used to smoke in the past decide to breastfeed. Age and when they left full-time education do not seem to have a major impact on the decision.

From the Null Deviance and the Residual Deviance we can calculate the X^2 statistic of the model:

X^2 = Null deviance - Residual deviance X^2 = 156.58 - 94.40 = 62.18

We have $p = 3$ predictor variables degrees of freedom. Generating a p-value from our chi-square score, we can see that the P-Value is < .00001. The result is significant at p < .05. Since the p-value is much less than .05, we can conclude that the model is useful for predicting the likelihood that a mother will breastfeed.

We will proceed fitting a logistic regression (LR) model through the library `tidymodels`.

Fitting GLM model with 'tidymodels'

We will define a generalized linear model for binary outcomes, specifying in *lr_spec* the package that we will use to fit the model ('glm' in this case), and setting the model's mode to classification. After the specification we will fit the model.

```
lr_spec <- logistic_reg() %>% # define a generalized linear model for binary outcomes
    set_engine("glm") %>% # declare which package will be used to fit the model
    set_mode("classification") # set model's mode to classification

lr_fit_tr <- lr_spec %>%
    fit(breast ~ pregnancy + howfed + howfedfr + partner + age
        + educat + ethnic + smokenow + smokebf,
        data=glm_train)

lr_fit_tr %>%
    pluck("fit") %>% # this function from the purrr library selects the "fit" slot
    summary()
```

```
##
## Call:
## stats::glm(formula = breast ~ pregnancy + howfed + howfedfr +
##     partner + age + educat + ethnic + smokenow + smokebf, family = stats::binomial,
##     data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1878  -0.2786   0.1888   0.4054   1.8912
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -8.10786    3.35032  -2.420  0.01552 *
## pregnancyBeginning -1.28967    0.75055  -1.718  0.08574 .
## howfedBreast        0.42938    0.77105   0.557  0.57761
## howfedfrBreast      0.90331    0.75349   1.199  0.23059
## partnerPartner      1.47743    0.86028   1.717  0.08591 .
## age                 0.06653    0.06444   1.032  0.30185
## educat              0.33488    0.17389   1.926  0.05413 .
## ethnicNon-white     2.40510    0.94144   2.555  0.01063 *
## smokenowYes        -3.90884    1.22958  -3.179  0.00148 **
## smokebfYes          1.77969    1.17336   1.517  0.12933
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 117.278  on 100  degrees of freedom
## Residual deviance:  63.113  on  91  degrees of freedom
## AIC: 83.113
##
## Number of Fisher Scoring iterations: 6
```

Now we are ready to get the predictions. We will use the `augment()` function that adds the predictions (labels and probabilities) to the original dataframe:

```
augment(lr_fit_tr, new_data=glm_test) %>% # this silently evaluates the model on glm_test
    conf_mat(truth=breast, estimate=.pred_class)
```

```
##           Truth
## Prediction Bottle Breast
##     Bottle      5      2
##     Breast      4     23
```

And then we can compute the accuracy with a Confusion Matrix:

```
# to compute the accuracy:
augment(lr_fit_tr, new_data=glm_test) %>%
    accuracy(truth=breast, estimate=.pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.824
```

Our model is 82% accurate, which is not bad at all. To improve the model, we can proceed by eliminating those variables that have minimal impact with respect to the choice to breastfeed. We try to remove the variables *age*, *educat* and *howfed*, since from our summary we found that they are the least impactful variables on response variable breast.

```
lr_fit_tr <- lr_spec %>%
    fit(breast ~ pregnancy + howfedfr + partner + ethnic + smokenow + smokebf,
        data=glm_train)

lr_fit_tr %>%
    pluck("fit") %>% # this function from the purrr library selects the "fit" slot
    summary()
```

```
##
## Call:
## stats::glm(formula = breast ~ pregnancy + howfedfr + partner +
##     ethnic + smokenow + smokebf, family = stats::binomial, data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1960  -0.3921   0.2049   0.4335   1.7200
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -0.9509     0.8725  -1.090  0.27578
## pregnancyBeginning  -0.7755     0.6460  -1.200  0.22995
## howfedfrBreast       1.3688     0.6323   2.165  0.03042 *
## partnerPartner       1.8994     0.7791   2.438  0.01477 *
## ethnicNon-white      2.3110     0.7988   2.893  0.00381 **
## smokenowYes         -3.2403     1.0638  -3.046  0.00232 **
## smokebfYes           1.0712     1.0207   1.049  0.29399
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 117.278  on 100  degrees of freedom
## Residual deviance:  69.857  on  94  degrees of freedom
## AIC: 83.857
##
## Number of Fisher Scoring iterations: 6
```

We generate the Confusion Matrix:

```
augment(lr_fit_tr, new_data=glm_test) %>%
    conf_mat(truth=breast, estimate=.pred_class)
```

```
##           Truth
## Prediction Bottle Breast
##     Bottle      5      1
##     Breast      4     24
```

And then let's compute the accuracy again:

```
augment(lr_fit_tr, new_data=glm_test) %>%
    accuracy(truth=breast, estimate=.pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.853
```

Indeed we can see that the accuracy of our model has increased from 82% to 85%.

Fitting K-NN classifier

Let's fit a k-NN classifier. Since k-NN involves calculating distances between datapoints, we must use numeric variables only. Therefore, it is needed to pre-process the data, transforming categorical variables into dummy variables.

We will make a copy of our data set so that we can prepare it for our k-NN classification.

```
breastfeed_for_knn <- processed_breastfeed
```

Next, we will put our outcome variable, *breast*, into its own object and remove it from the data set.

```
# we put the outcome in its own object
breast_outcome <- breastfeed_for_knn %>% select(breast)

# remove original variable from the data set
breastfeed_for_knn <- breastfeed_for_knn %>% select(-breast)
```

The outcome variable for k-NN classification should remain a factor variable.

Now we are ready to dummy code any factor or categorical variables.

```
str(breastfeed_for_knn)
```

```
## 'data.frame':    135 obs. of  9 variables:
##  $ pregnancy: Factor w/ 2 levels "End","Beginning": 2 2 2 2 2 2 2 2 2 2 ...
##  $ howfed   : Factor w/ 2 levels "Bottle","Breast": 2 1 2 2 2 2 2 2 2 2 ...
##  $ howfedfr : Factor w/ 2 levels "Bottle","Breast": 2 2 2 2 2 2 2 1 2 2 ...
##  $ partner  : Factor w/ 2 levels "Single","Partner": 2 2 2 2 2 2 2 2 2 2 ...
##  $ smokenow : Factor w/ 2 levels "No","Yes": 1 1 1 2 1 1 1 2 1 1 ...
##  $ smokebf  : Factor w/ 2 levels "No","Yes": 1 1 1 2 1 1 1 2 2 1 ...
##  $ age      : num  24 27 39 29 21 27 27 20 31 28 ...
##  $ educat   : num  19 18 16 16 21 19 22 19 17 16 ...
##  $ ethnic   : Factor w/ 2 levels "White","Non-white": 2 1 1 1 1 2 2 2 1 1 ...
##  - attr(*, "na.action")= 'omit' Named int [1:4] 6 22 46 125
##   ..- attr(*, "names")= chr [1:4] "6" "22" "46" "125"
```

We already know that pregnancy, howfed, howfedfr, partner, smokenow, smokebf and ethnic are all factor variables that have only two levels. We can dummy code them, in order to have variables with just two levels and coded 1/0.

```
breastfeed_for_knn$pregnancy <- dummy.code(breastfeed_for_knn$pregnancy)
breastfeed_for_knn$howfed <- dummy.code(breastfeed_for_knn$howfed)
breastfeed_for_knn$howfedfr <- dummy.code(breastfeed_for_knn$howfedfr)
breastfeed_for_knn$partner <- dummy.code(breastfeed_for_knn$partner)
breastfeed_for_knn$smokenow <- dummy.code(breastfeed_for_knn$smokenow)
breastfeed_for_knn$smokebf <- dummy.code(breastfeed_for_knn$smokebf)
breastfeed_for_knn$ethnic <- dummy.code(breastfeed_for_knn$ethnic)
```

Splitting the data into training and test sets for k-NN

As we did in our GLM model, we proceed with the split into train and test set.

```
# creating test and training sets that contain all of the predictors
class_pred_train <- breastfeed_for_knn[train_ind, ]
class_pred_test <- breastfeed_for_knn[-train_ind, ]
```

We will also split the response variable into training and test sets using the same partition.

```
breast_outcome_train <- breast_outcome[train_ind, ]
breast_outcome_test <- breast_outcome[-train_ind, ]
```

Running K-NN Classification

Using the k-NN algorithm we must be precise in using the right number of k. For this reason we will initially make three attempts, with k equal to 1, 3, and 5, respectively. Finally we will try the *caret* package, which automatically chooses the optimal number of k.

Model evaluation with k=1

We perform the fit, specifying the training and test sets and the factor of true classifications of training sets. Initially k will be equal to 1. The model is saved in *breast_pred_knn*.

```
breast_pred_knn <- knn(train = class_pred_train, test = class_pred_test,
                       cl = breast_outcome_train, k=1)
```

Next we can make an evaluation of the model. We create a dataframe from the test variable *breast_outcome_test*, merge it with the dataframe of the model we created earlier, *breast_pred_knn*, and then compare the predicted breast variable outcomes with the observed ones.

```
# put "breast_outcome_test" in a data frame
breast_outcome_test <- data.frame(breast_outcome_test)

# merge "breast_pred_knn" and "breast_outcome_test"
class_comparison <- data.frame(breast_pred_knn, breast_outcome_test)

# specify column names for "class_comparison"
names(class_comparison) <- c("PredictedBreast", "ObservedBreast")

# inspect "class_comparison"
head(class_comparison)
```

```
##   PredictedBreast ObservedBreast
## 1          Breast         Breast
## 2          Breast         Breast
## 3          Breast         Breast
## 4          Breast         Breast
## 5          Bottle         Bottle
## 6          Breast         Breast
```

We can create a table to examine the model accuracy:

```
CrossTable(x = class_comparison$ObservedBreast, y = class_comparison$PredictedBreast,
           prop.chisq=FALSE, prop.c = FALSE, prop.r = FALSE, prop.t = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  34
##
##
##                                 | class_comparison$PredictedBreast
## class_comparison$ObservedBreast |    Bottle |    Breast | Row Total |
## --------------------------------|-----------|-----------|-----------|
##                          Bottle |         4 |         5 |         9 |
## --------------------------------|-----------|-----------|-----------|
##                          Breast |         3 |        22 |        25 |
## --------------------------------|-----------|-----------|-----------|
##                    Column Total |         7 |        27 |        34 |
## --------------------------------|-----------|-----------|-----------|
##
##
```

Our model did not perform very well. The diagonal of the matrix represents the number of cases that were correctly classified for each category. If the model correctly classified all cases, the matrix would have zeros everywhere but the diagonal. Let's increase our number of k to 3.

Model evaluation k=3

We repeat all previous steps, this time by tuning the number of k to 3:

```
breast_pred_knn <- knn(train = class_pred_train, test = class_pred_test,
                       cl = breast_outcome_train, k=3)
```

Model evaluation:

```
breast_outcome_test <- data.frame(breast_outcome_test)

class_comparison <- data.frame(breast_pred_knn, breast_outcome_test)

names(class_comparison) <- c("PredictedBreast", "ObservedBreast")

head(class_comparison)
```

```
##   PredictedBreast ObservedBreast
## 1          Breast         Breast
## 2          Breast         Breast
## 3          Breast         Breast
## 4          Breast         Breast
## 5          Breast         Bottle
## 6          Breast         Breast
```

Generating the CrossTable:

```
CrossTable(x = class_comparison$ObservedBreast, y = class_comparison$PredictedBreast,
           prop.chisq=FALSE, prop.c = FALSE, prop.r = FALSE, prop.t = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  34
##
##
##                                  | class_comparison$PredictedBreast
## class_comparison$ObservedBreast |    Bottle |    Breast | Row Total |
## --------------------------------|-----------|-----------|-----------|
##                          Bottle |         3 |         6 |         9 |
## --------------------------------|-----------|-----------|-----------|
##                          Breast |         2 |        23 |        25 |
## --------------------------------|-----------|-----------|-----------|
##                    Column Total |         5 |        29 |        34 |
## --------------------------------|-----------|-----------|-----------|
##
##
```

We continue to increase the number of k to 5.

Model evaluation k=5

Fitting the kNN with k = 5:

```
breast_pred_knn <- knn(train = class_pred_train, test = class_pred_test,
                       cl = breast_outcome_train, k=5)
```

Model evaluation:

```
# put "breast_outcome_test" in a data frame
breast_outcome_test <- data.frame(breast_outcome_test)

# merge "breast_pred_knn" and "breast_outcome_test"
class_comparison <- data.frame(breast_pred_knn, breast_outcome_test)

# specify column names for "class_comparison"
names(class_comparison) <- c("PredictedBreast", "ObservedBreast")

# inspect "class_comparison"
head(class_comparison)
```

```
##   PredictedBreast ObservedBreast
## 1          Breast         Breast
## 2          Breast         Breast
## 3          Breast         Breast
## 4          Breast         Breast
## 5          Breast         Bottle
## 6          Breast         Breast
```

CrossTable:

```
CrossTable(x = class_comparison$ObservedBreast, y = class_comparison$PredictedBreast,
           prop.chisq=FALSE, prop.c = FALSE, prop.r = FALSE, prop.t = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |-------------------------|
##
##
## Total Observations in Table:  34
##
##
```

```
##                                  | class_comparison$PredictedBreast 
## class_comparison$ObservedBreast |    Bottle |    Breast | Row Total | 
## --------------------------------|-----------|-----------|-----------|
##                          Bottle |         2 |         7 |         9 | 
## --------------------------------|-----------|-----------|-----------|
##                          Breast |         2 |        23 |        25 | 
## --------------------------------|-----------|-----------|-----------|
##                    Column Total |         4 |        30 |        34 | 
## --------------------------------|-----------|-----------|-----------|
## 
## 
```

The k-NN model appears not to perform as well as the logistic regression model.

k-NN classifier with caret package

Now we will try with caret package, that will pick the optical number of neighbors automatically:

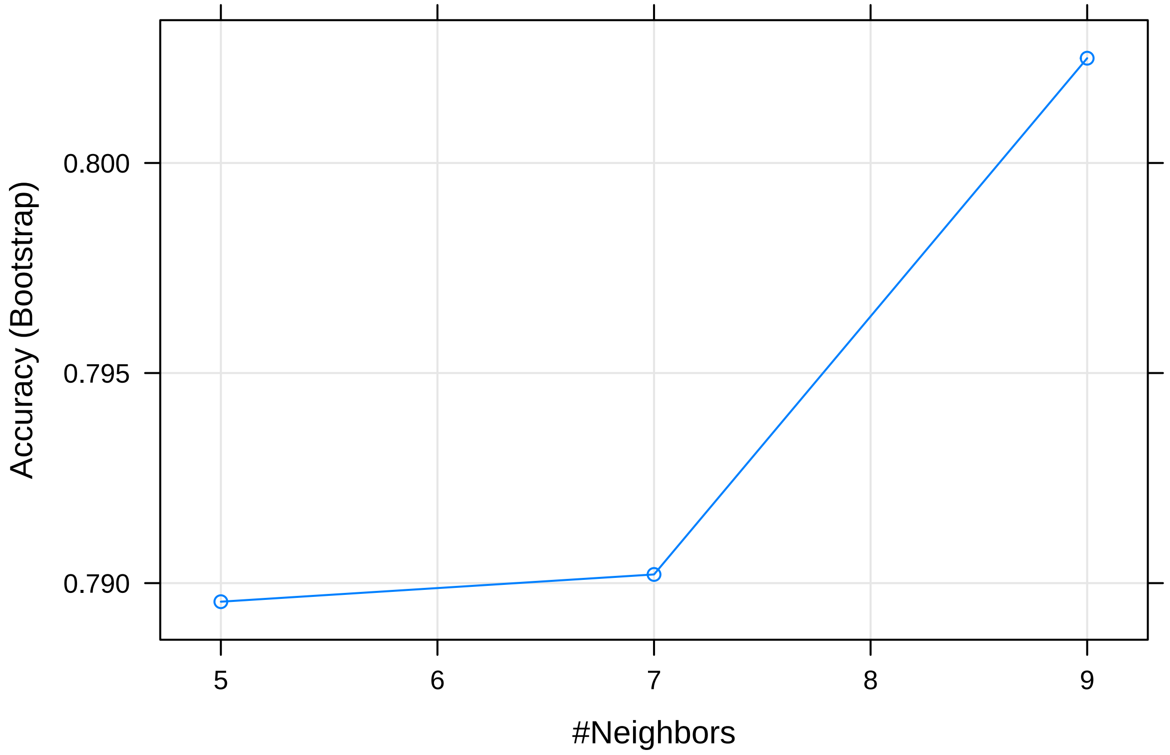
```
breast_pred_caret <- train(class_pred_train, breast_outcome_train,
                           method = "knn", preProcess = c("center","scale"))

breast_pred_caret
```

```
## k-Nearest Neighbors 
## 
## 101 samples
##   9 predictor
##   2 classes: 'Bottle', 'Breast' 
## 
## Pre-processing: centered (2), scaled (2), ignore (7) 
## Resampling: Bootstrapped (25 reps) 
## Summary of sample sizes: 101, 101, 101, 101, 101, 101, ... 
## Resampling results across tuning parameters:
## 
##   k  Accuracy   Kappa    
##   5  0.7895581  0.4352289
##   7  0.7902063  0.4418849
##   9  0.8024937  0.4662623
## 
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

As the output explains, parameter tuning was performed based on the results ti accuracy. The final value that was used for the model is a k equal to 9.

Let's plot a graph just to see the difference in accuracy of k between 5, 7, and 9:

```
plot(breast_pred_caret)
```



As the graph shows us, the accuracy goes up from a minimum of 78% to a maximum of 80%. That still remains worse than the accuracy of the logistic regression model.

Fitting the Naïve Bayes classifier

The standard Naïve Bayes classifier assumes independence of the predictor variables, and Gaussian distribution (given the target class) of metric predictors.

Let's fit the model with the formula used for logit, specifying the dataset (*processed_breastfeed*) and the positive double controlling Laplace smoothing.

```
naivebayes_model <- naiveBayes(breast ~ pregnancy + howfed + howfedfr + partner + age
                               + educat + ethnic + smokenow + smokebf,
                               data = processed_breastfeed,
                               laplace = 3)
```

Then we store the predictions and we generate a Confusion Matrix:

```
pred <- predict(naivebayes_model, processed_breastfeed[,-1])
cMatrix <- table(pred, processed_breastfeed$breast)
confusionMatrix(cMatrix,
                positive="Breast")
```

```
## Confusion Matrix and Statistics
##
##
## pred     Bottle Breast
##   Bottle     24     13
##   Breast     12     86
##
##                  Accuracy : 0.8148
##                    95% CI : (0.7389, 0.8764)
##       No Information Rate : 0.7333
##       P-Value [Acc > NIR] : 0.01779
##
```

```
##                   Kappa : 0.5307
##
##  Mcnemar's Test P-Value : 1.00000
##
##             Sensitivity : 0.8687
##             Specificity : 0.6667
##          Pos Pred Value : 0.8776
##          Neg Pred Value : 0.6486
##              Prevalence : 0.7333
##          Detection Rate : 0.6370
##    Detection Prevalence : 0.7259
##       Balanced Accuracy : 0.7677
##
##        'Positive' Class : Breast
##
```

This model achieves an accuracy of 81%.

Conclusions

Given the performance of the various methods, we are now ready for a comparison of the results.

Target audience for breastfeeding promotion

We have already discussed the results of the logistic regression model summary. Thus, we can assume that if we need to target breastfeeding promotions toward women with a lower probability of choosing it, we would need to target women who are early in their pregnancy and whose friendships are not inclined toward breastfeeding. Non-white ethnic women without partners are more likely not to breastfeed, so we might refer to them as well. Smoking women veer firmly toward bottle-feeding, so certainly they can be included in the target audience.

Models performance

In terms of performance, we found that:

- the *logistic regression model* achieves an accuracy of 85%;
- the *kNN classifier,* after tuning the k parameters (the optimal one is 9), reaches an accuracy of 80%;
- the *Naïve Bayes classifier* reaches an accuracy of about 81%.

We can therefore conclude that *the model that performs best on this dataset is the logistic regression model.*