HarvardX PH125.9 Cancer Project

Alexander Braune

08.03.2022

Abstract

This project shows basis steps of exploratory data analysis and the building basic classification machine learning models: Logarithmical Regression, Random Forrest and Support vector machines. As a use case the UCI Cervical Cancer data set is used to construct a forecast method that predicts classification of risk facotor constellations into cancerous and non-cancerous. The final SVM model provides a validated accuracy of .87 and comparably lacks Specificity the least of all three models.

Contents

1 Introduction and Overview

Medical classifications are a well known field where Data Science techniques are applied. The increased use of Machine Learning algorithms for predicting diagnostic findings adds value to healthcare in a way that goes beyond descriptive statistics. Predictive data analytics supports medical staff when determining a diagnosis or act as a warning mechanism as the importance of variables is revealed for positive cases and variables with a high importance can be designated as risk factors.

Especially cancer research is heavily focused in early predictions, therefore the revealing of risk factors and classification into positive and negative cases from variable inputs is a common use case for Healthcare Data Scientists.

Because of this real world applicability the development of a Machine Learning algorithm that predicts cancer indications will be the aim of this final project for the edX course PH125.9x form HarvardX.

The data used is the *"Cervical cancer (Risk Factors) Data Set"* offered by the UCI Machine Learning Repositor (published 03/2017). The dataset was donated from research at the "Hospital Universitario de Caracas" in Venezuela. It comprises demographic information, habits, and historic medical records of 858 patients.

The records are organized per patient and contains 30 columns of variables that describe the mediacal profile of the patient. The four target columns contain the result of different cervical cancer indications: Hinselmann, Schiller, Cytology and Biopsy.

The Model is therefore set up to classify a given record set of variables from a new patient into the classes *"0 cancer indications"* and *"1 or more cancer indications*".

In chapter 2.1 the data format and logic is inspected, missing values are imputed and dataframes are prepared for model input.

Following in chapter 2.2, the steps of exploratory data analysis offer an understanding of the input variables and target variables. The data correlation, distribution of input and target variables are inspected and insights from variable constellations are generated.

Model training is done with partitioned data from chapter 2.3.

The insights generated are then used in chapter 2.4 to build three common Machine Learning models for classification from scratch (*Logarithmical Regression*, *Random Forrest Prediction* and *Support Vector Machines*).
The models are trained and compared by evaluating the the overall accuracy. All models are then evaluated with the validation data set in chapter 3 from which are conclusions drawn in chapter 4.

2 Methods and Analysis

In these sections the data is prepared, explored, partitioned and the steps of model building and parametrization are described.

2.1 Data Preperation

The dataset is downloaded from https://archive.ics.uci.edu/ml/machine-learning-databases/00383/risk_factors_cervical_cancer.csv. It is read in with basic R commands as it is not zipped but a plain comma separated value (csv) file and stored as a dataframe.

To investigate the download the header and column classes of the dataframe are evaluated:

```
##                                    [,1]
## Age                                "integer"
## Number.of.sexual.partners          "character"
## First.sexual.intercourse           "character"
## Num.of.pregnancies                 "character"
## Smokes                             "character"
## Smokes..years.                     "character"
## Smokes..packs.year.                "character"
## Hormonal.Contraceptives            "character"
## Hormonal.Contraceptives..years.    "character"
## IUD                                "character"
## IUD..years.                        "character"
## STDs                               "character"
## STDs..number.                      "character"
## STDs.condylomatosis                "character"
## STDs.cervical.condylomatosis       "character"
## STDs.vaginal.condylomatosis        "character"
## STDs.vulvo.perineal.condylomatosis "character"
## STDs.syphilis                      "character"
## STDs.pelvic.inflammatory.disease   "character"
## STDs.genital.herpes                "character"
## STDs.molluscum.contagiosum         "character"
## STDs.AIDS                          "character"
## STDs.HIV                           "character"
## STDs.Hepatitis.B                   "character"
## STDs.HPV                           "character"
## STDs..Number.of.diagnosis          "integer"
## STDs..Time.since.first.diagnosis   "character"
## STDs..Time.since.last.diagnosis    "character"
## Dx.Cancer                          "integer"
## Dx.CIN                             "integer"
## Dx.HPV                             "integer"
## Dx                                 "integer"
## Hinselmann                         "integer"
## Schiller                           "integer"
## Citology                           "integer"
## Biopsy                             "integer"
```

It can be noted that most columns are string, although numerics are expected. With head() a preview of the first rows is generated to inspect why read.csv() created columns as a string class.

	1	2	3	4	5	6
Age	18	15	34	52	46	42
Number.of.sexual.partners	4.0	1.0	1.0	5.0	3.0	3.0
First.sexual.intercourse	15.0	14.0	?	16.0	21.0	23.0
Num.of.pregnancies	1.0	1.0	1.0	4.0	4.0	2.0
Smokes	0.0	0.0	0.0	1.0	0.0	0.0
Smokes..years.	0.0	0.0	0.0	37.0	0.0	0.0
Smokes..packs.year.	0.0	0.0	0.0	37.0	0.0	0.0
Hormonal.Contraceptives	0.0	0.0	0.0	1.0	1.0	0.0
Hormonal.Contraceptives..years.	0.0	0.0	0.0	3.0	15.0	0.0
IUD	0.0	0.0	0.0	0.0	0.0	0.0
IUD..years.	0.0	0.0	0.0	0.0	0.0	0.0
STDs	0.0	0.0	0.0	0.0	0.0	0.0
STDs..number.	0.0	0.0	0.0	0.0	0.0	0.0
STDs.condylomatosis	0.0	0.0	0.0	0.0	0.0	0.0
STDs.cervical.condylomatosis	0.0	0.0	0.0	0.0	0.0	0.0
STDs.vaginal.condylomatosis	0.0	0.0	0.0	0.0	0.0	0.0
STDs.vulvo.perineal.condylomatosis	0.0	0.0	0.0	0.0	0.0	0.0
STDs.syphilis	0.0	0.0	0.0	0.0	0.0	0.0
STDs.pelvic.inflammatory.disease	0.0	0.0	0.0	0.0	0.0	0.0
STDs.genital.herpes	0.0	0.0	0.0	0.0	0.0	0.0
STDs.molluscum.contagiosum	0.0	0.0	0.0	0.0	0.0	0.0
STDs.AIDS	0.0	0.0	0.0	0.0	0.0	0.0
STDs.HIV	0.0	0.0	0.0	0.0	0.0	0.0
STDs.Hepatitis.B	0.0	0.0	0.0	0.0	0.0	0.0
STDs.HPV	0.0	0.0	0.0	0.0	0.0	0.0
STDs..Number.of.diagnosis	0	0	0	0	0	0
STDs..Time.since.first.diagnosis	?	?	?	?	?	?
STDs..Time.since.last.diagnosis	?	?	?	?	?	?
Dx.Cancer	0	0	0	1	0	0
Dx.CIN	0	0	0	0	0	0
Dx.HPV	0	0	0	1	0	0
Dx	0	0	0	0	0	0
Hinselmann	0	0	0	0	0	0
Schiller	0	0	0	0	0	0
Citology	0	0	0	0	0	0
Biopsy	0	0	0	0	0	0

In the preview strings of "?" are made visible. As numeric variable columns are therefore set as character fields, the "?" and " " string records are extracted and set as NA value entries.

```
#replace strings with NA
data <- na_if(data, "?")
data <- na_if(data, "")
```

The variables can then be transformed as numeric fields without warnings:

```
#convert to numeric columns
data <- as.data.frame(apply(data, 2, as.numeric))
```

Missing values can not be used when training regression models. As many variables should be integers (that indicate binary results) imputation with averages is not sufficient. NA values are therefore filled with a more robust method of filling values up with column medians.

```
# replace NA with median values of the columns
for(i in 1:32) {
  data[ , i][is.na(data[ , i])] <- median(data[ , i], na.rm = TRUE)
}
```

The consistency is therefore given as NA values are imputed with medians:

```
#check for na
anyNA(data)
```

```
## [1] FALSE
```

With colSums() empty columns only containing 0 values are found.

```
#check for empty columns
colSums(data, na.rm = T)
```

```
##                                Age          Number.of.sexual.partners
##                         23012.0000                          2155.0000
##           First.sexual.intercourse                 Num.of.pregnancies
##                         14582.0000                          1937.0000
##                             Smokes                     Smokes..years.
##                           123.0000                          1030.6646
##               Smokes..packs.year.            Hormonal.Contraceptives
##                           382.9066                           589.0000
##     Hormonal.Contraceptives..years.                                IUD
##                          1746.3144                            83.0000
##                          IUD..years.                               STDs
##                             381.4700                            79.0000
##                        STDs..number.                STDs.condylomatosis
##                             133.0000                            44.0000
##       STDs.cervical.condylomatosis        STDs.vaginal.condylomatosis
##                               0.0000                             4.0000
## STDs.vulvo.perineal.condylomatosis                      STDs.syphilis
##                              43.0000                            18.0000
##    STDs.pelvic.inflammatory.disease                STDs.genital.herpes
##                               1.0000                             1.0000
##         STDs.molluscum.contagiosum                          STDs.AIDS
##                               1.0000                             0.0000
##                             STDs.HIV                   STDs.Hepatitis.B
##                              18.0000                             1.0000
##                             STDs.HPV          STDs..Number.of.diagnosis
##                               2.0000                            75.0000
##    STDs..Time.since.first.diagnosis    STDs..Time.since.last.diagnosis
##                            3584.0000                          2774.0000
##                           Dx.Cancer                             Dx.CIN
##                              18.0000                             9.0000
##                               Dx.HPV                                 Dx
##                              18.0000                            24.0000
##                           Hinselmann                           Schiller
##                              35.0000                            74.0000
##                             Citology                             Biopsy
##                              44.0000                            55.0000
```

Empty columns have no value for prediction models and are dropped.

```r
#drop empty columns
i <- (colSums(data, na.rm=T) !=0)
data <- data[, i]
```

For the target values, first the count of positive indications is added:

```r
# create indications counter columns
data$cancer_indications = data$Schiller+data$Hinselmann+data$Citology+data$Biopsy
```

With a conservative approach, also having only *one* indication will be labeled as a positive cancer case.

```r
# signal cancer if at least one indication is found (1 for yes, 0 for no)
data$cancer[data$cancer_indications <1] <- 0
data$cancer[data$cancer_indications >0] <- 1
```

The input variables are extracted from the target variables to feed models for training with dataframes containing only input variables:

```
# extract variables subset without results
data_var<-data[c(1:30)]
```

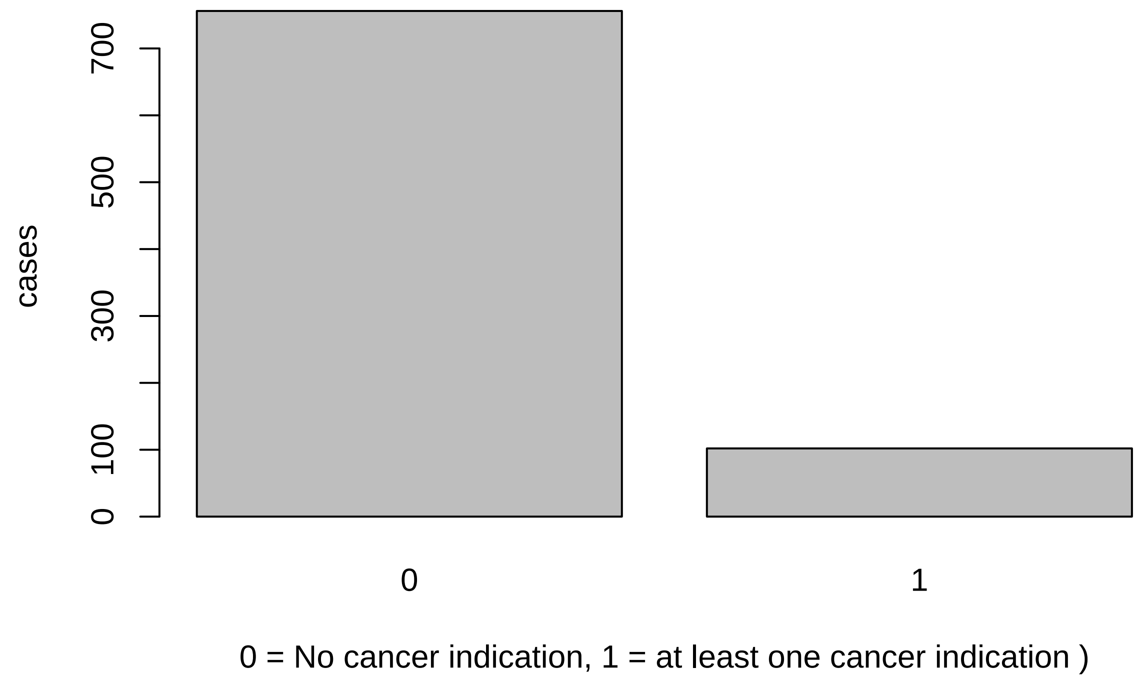
And the actual outcomes are extracted as y_act and stored as a factor column.

```
# extract actual results subset
y_act <- as.factor(data$cancer)
```

2.2 Data Analysis

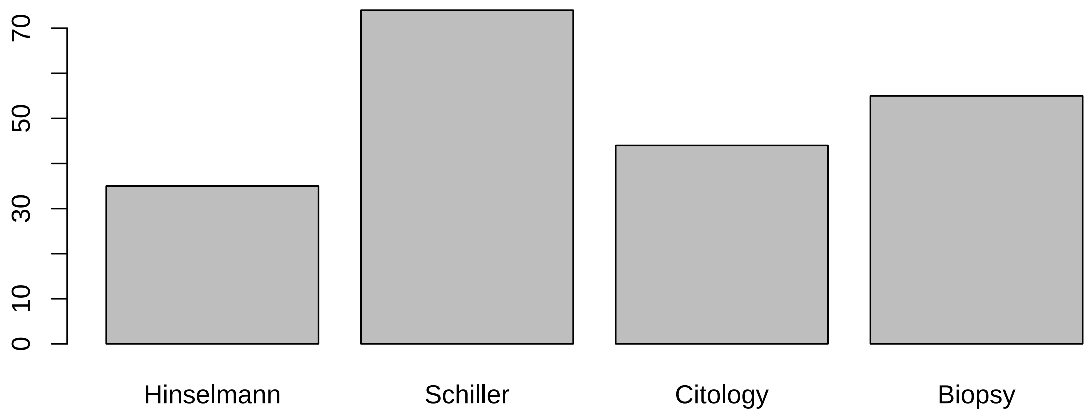
First, the target columns are evaluated, following the distribution and correlation of risk factor variables.

The created actual outcome column shows that 102 positive cancer cases are among the 858 patiens.

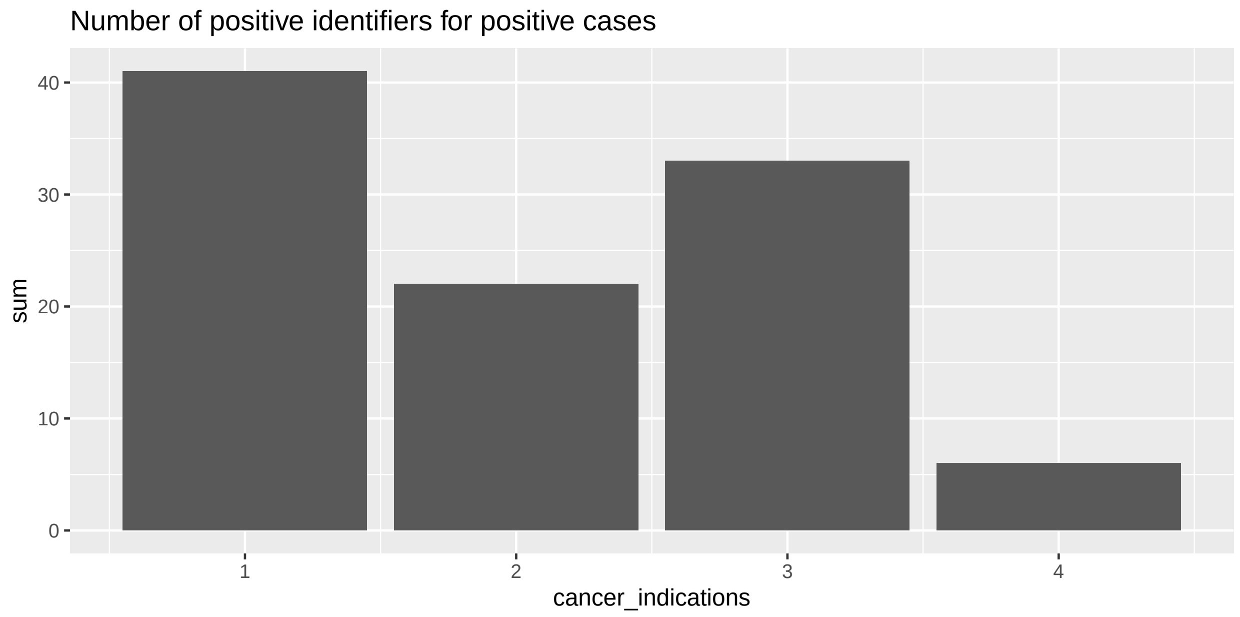


The random baseline guess is therefore that a new patient has a positive cancer indication with a probability of ~ 12%.

```
## Hinselmann   Schiller   Citology     Biopsy
##         35         74         44         55
```

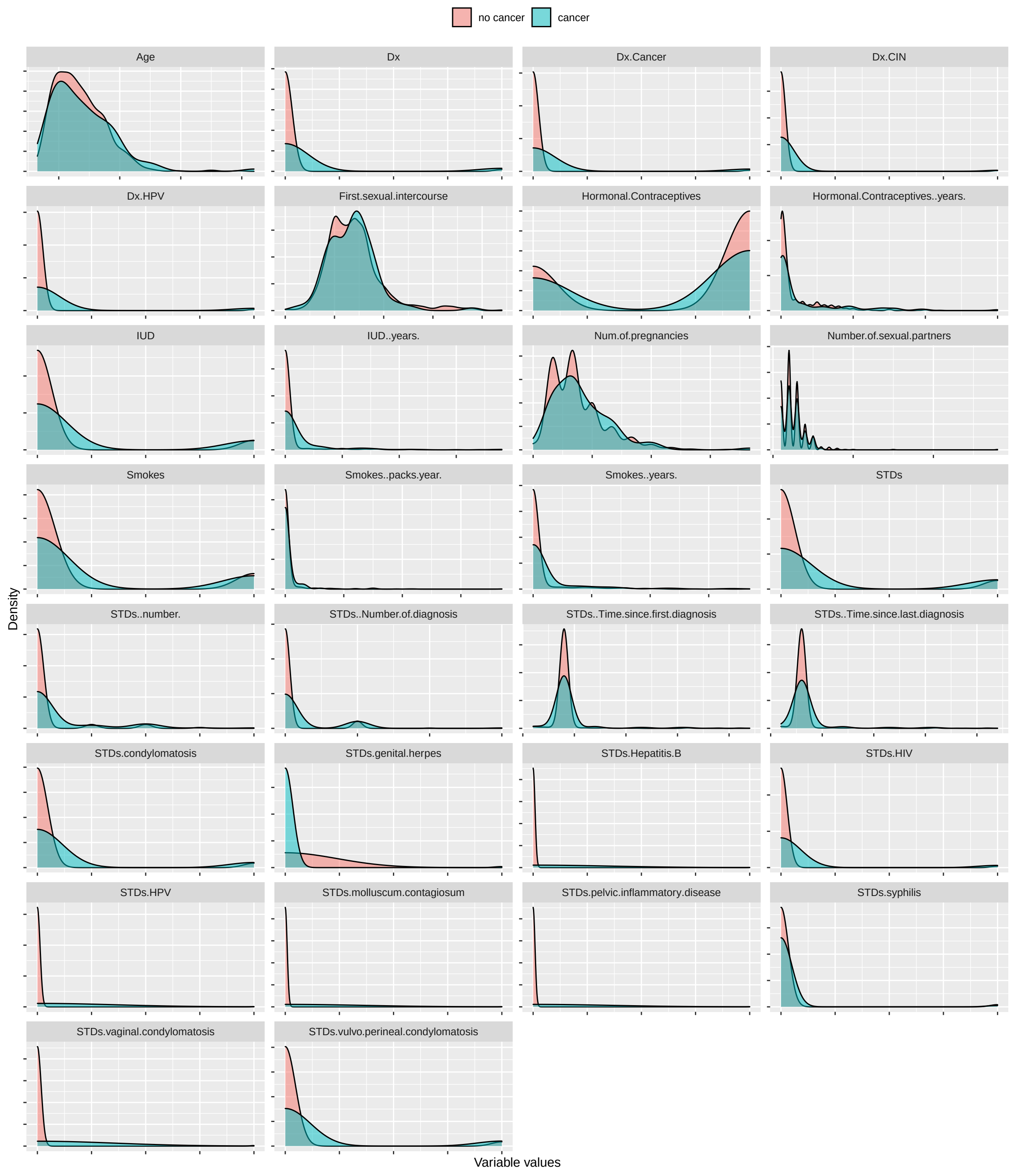


With 74 cases Schiller is the most sensitive indicator and Hinselmann with 35 the least reactive. In combintation this furthers the insight that the overlapping indications are not broad.

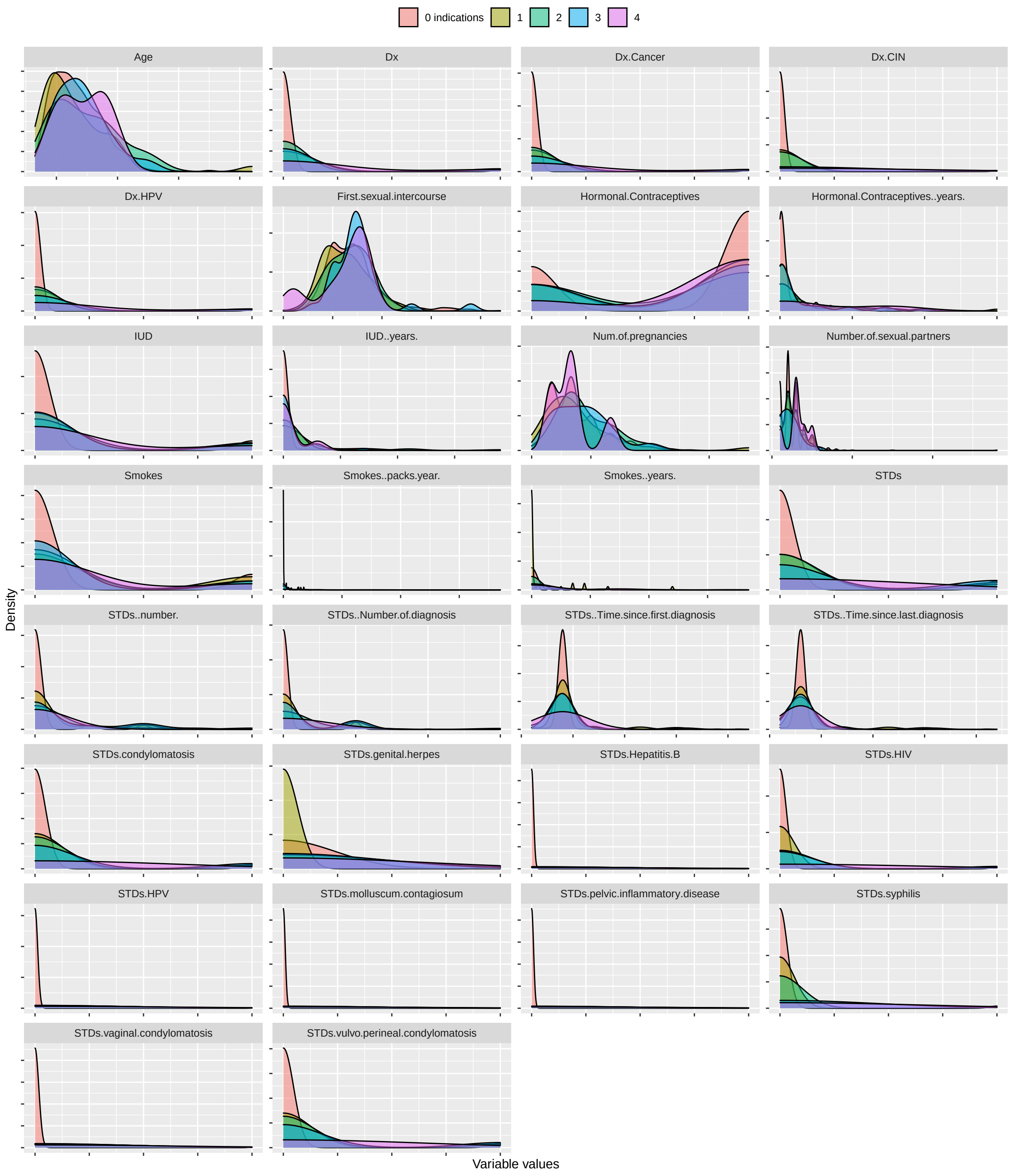


Only 6 cases have all four identifier. It is therefore accurate to assume a positive indication with at least one identifier is the broader and more conservative definition of [cancer=1].

Following the risk factor variables are plotted with the layers [cancer=1] and [cancer=0].



It is visible that the distribution of cancerous and non-cancerous cases is around a similar average, but some risk factors have a higher density over the average. For our models this means that the constellation of variables will important as well. Also, variables with a more distinct profile between [cancer=0] and [cancer=1] should be more valuable for model training.

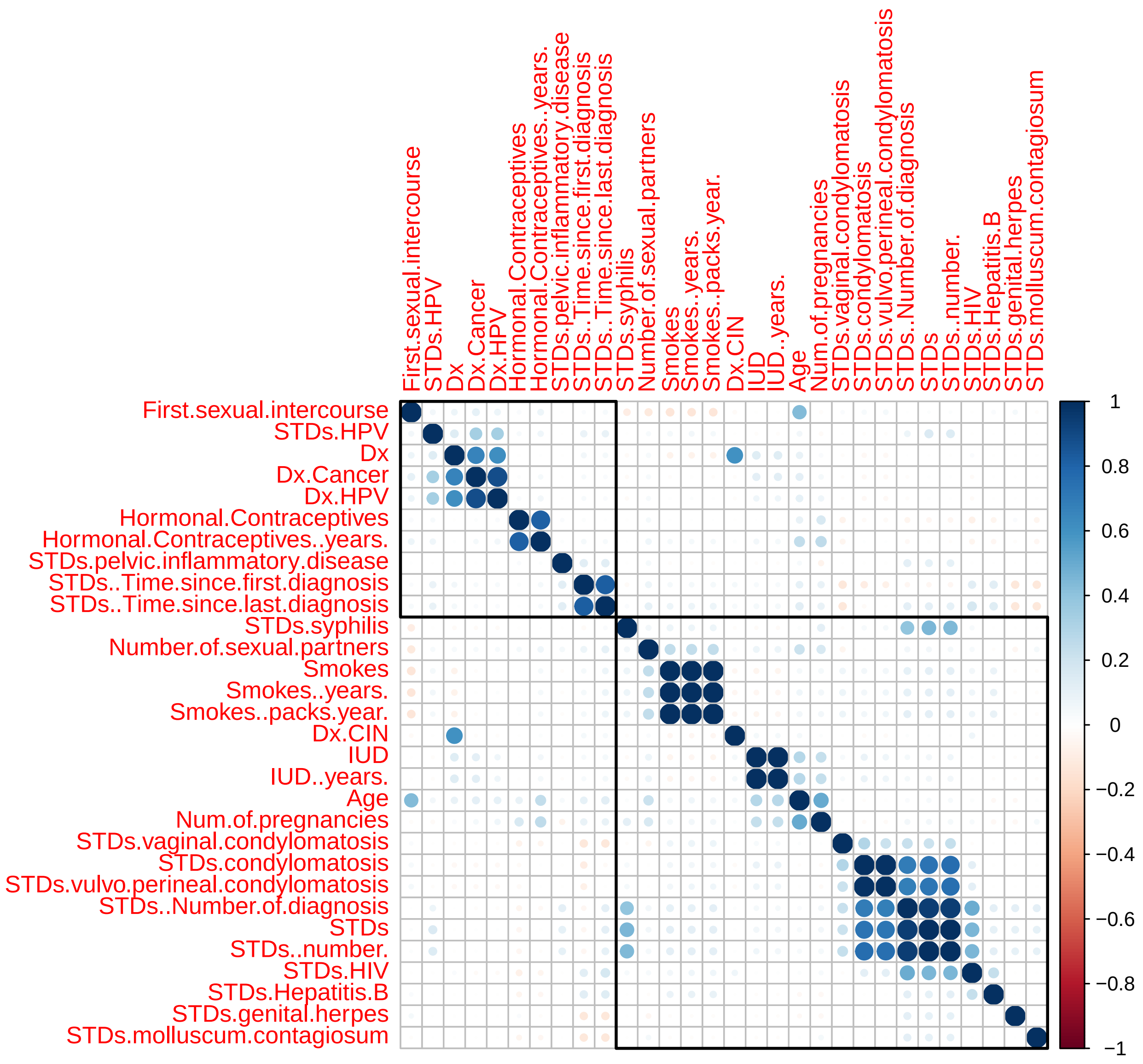


The differentiation by indication count shows that some variables have a high density above a single value. This leads to the conclusion this column has a low variability. These add low value to the prediction models and are eliminated in a near-zero-variability subset, like the “empty” columns with 0 values only.

```r
# exclude near zero variance columns
data_var_nzv <- data_var[,-nearZeroVar(data_var)]
```

	1	2	3	4	5	6
Age	18	15	34	52	46	42
Number.of.sexual.partners	4	1	1	5	3	3
First.sexual.intercourse	15	14	17	16	21	23
Num.of.pregnancies	1	1	1	4	4	2
Smokes	0	0	0	1	0	0
Smokes..years.	0	0	0	37	0	0
Smokes..packs.year.	0	0	0	37	0	0
Hormonal.Contraceptives	0	0	0	1	1	0
Hormonal.Contraceptives..years.	0	0	0	3	15	0
IUD	0	0	0	0	0	0
IUD..years.	0	0	0	0	0	0
STDs	0	0	0	0	0	0
STDs..number.	0	0	0	0	0	0
STDs.condylomatosis	0	0	0	0	0	0
STDs.vaginal.condylomatosis	0	0	0	0	0	0
STDs.vulvo.perineal.condylomatosis	0	0	0	0	0	0
STDs.syphilis	0	0	0	0	0	0
STDs.pelvic.inflammatory.disease	0	0	0	0	0	0
STDs.genital.herpes	0	0	0	0	0	0
STDs.molluscum.contagiosum	0	0	0	0	0	0
STDs.HIV	0	0	0	0	0	0
STDs.Hepatitis.B	0	0	0	0	0	0
STDs.HPV	0	0	0	0	0	0
STDs..Number.of.diagnosis	0	0	0	0	0	0
STDs..Time.since.first.diagnosis	4	4	4	4	4	4
STDs..Time.since.last.diagnosis	3	3	3	3	3	3
Dx.Cancer	0	0	0	1	0	0
Dx.CIN	0	0	0	0	0	0
Dx.HPV	0	0	0	1	0	0
Dx	0	0	0	0	0	0
Hinselmann	0	0	0	0	0	0
Schiller	0	0	0	0	0	0
Citology	0	0	0	0	0	0
Biopsy	0	0	0	0	0	0
cancer_indications	0	0	0	0	0	0
cancer	0	0	0	0	0	0

Following the variable correlation (dependencies between input values) is inspected:



Cluster of highly correlated variables around Smoke and STD can be graphically derived. There add also low value to the model training. For example *"Smokes packs per year"* and *"Smokes years"* are highly correlated. Considering both therefore only adds noise. Highly correlated variables with a correlation above .75 are:

```
## [1] "STDs..number."                     "STDs"
## [3] "STDs.condylomatosis"               "Smokes..years."
## [5] "Smokes..packs.year."               "STDs..Time.since.last.diagnosis"
## [7] "Dx.Cancer"                         "IUD"
## [9] "Hormonal.Contraceptives..years."
```

2.3 Data Partitioning

Before applying any data investigation the whole data set is split into a 70% training set and a 30% validation set. The count of records prevents the validation set to be smaller.

A model that is parametrized with the whole data set is prone to overfitting and one can not determine a true forecast accuracy as all data is already known to the model (bias).

With randomly splitting the data into separate sets the training of the model can be done without overfitting it. After all parametrization of the model the true accuracy can be evaluated with a previously unconsidered validation data set.

```
#partition datasets, 70% for training, 30% for validation
set.seed(2, sample.kind="Rounding")
test_index <- createDataPartition(y = data$Age, times = 1, p = 0.3, list = FALSE)

# split variable dataset
train_var <- data_var[-test_index,]
validate_var <- data_var[test_index,]

# split results dataset
train_y_act <- as.factor(data[-test_index,36])
validate_y_act <- as.factor(data[test_index,36])

# split near sero variance dataset
train_var_nzv <- data_var_nzv[-test_index,]
validate_var_nzv <- data_var_nzv[test_index,]
```

Both datasets are checked with their dimensions:

```
## # rows traing:  598
```

```
## # fields training:  30
```

```
## # rows validation:  260
```

```
## # fields validation:  30
```

2.4 Modelling Approach

First, a simple logistical regression model is trained with the training subset. The logistically transformed regression provides probabilities for a continous outcome. These probability needs to be split into the binary classification outcome. The translated outcome is compared to the actual training outcomes to provide the model accuracy within the training sample. The Logistical Regression does not converge with the whole dataset and is therefore trained with the excluded variables of low variability.

```
# create log regression model
set.seed(192)
model1 <- glm(train_y_act ~ . , data=train_var_nzv, family = "binomial", maxit = 25)

# predict responses with trained log function
y_fc1_glm = predict(model1, newdata = train_var_nzv, type = 'response')

# translate into binary results of cancer
y_fc1 <- factor(ifelse(y_fc1_glm > 0.5, 1, 0))

# model evaluation
confusionMatrix(train_y_act, y_fc1)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8812709
```

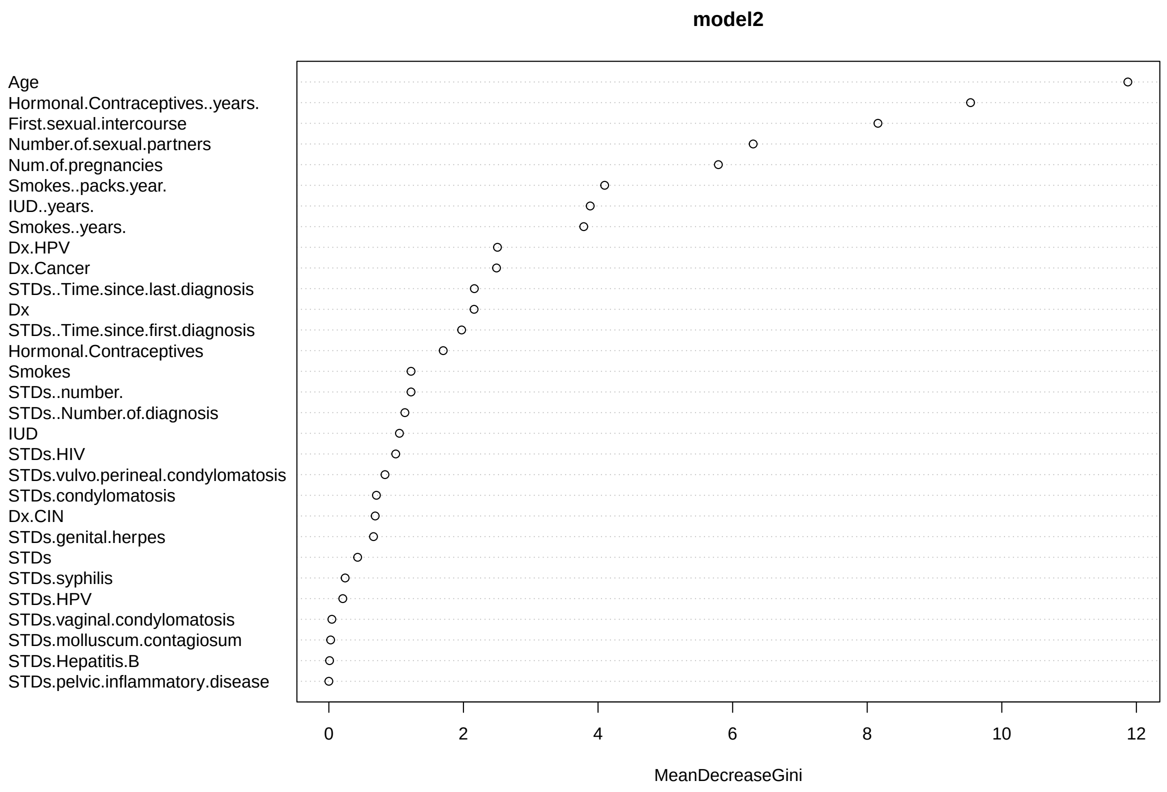
The logistic regression is dependend on the nzv subset of data, but with this preparation a high overall accuracy seems achievable.

The second model is a Random Forrest generator, which sets up decision trees in a random fashion. The algorithm creates connected node weights and compares outcomes and therefore randomly finds decision trees which classify the records with a higher accuracy as the random sample.

```
#RandomForest creation
set.seed(392)
model2 = randomForest(x = train_var,
                      y = train_y_act,
                      ntree = 100)
# write results
y_fc2 = predict(model2, newdata = train_var, type = 'class')

# model evaluation
confusionMatrix(train_y_act, y_fc2)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9280936
```



The importance of variables (high meanDecreaseGini) reveals that in a pareto manner a quintile of variables has the highest contribution in prediction impact.

As a third model, the Support Vector Machine model aims to be efficient and needs less computational power. The algorithm looks for hyerplanes in the n-variable-dimensional space. The classification is done by looking for a vector that is the best distinction between variables from both classes:

```r
#SVM creation
set.seed(592)
model3 = svm(formula = train_y_act ~ .,
                  data = train_var,
                  type = 'C-classification',
                  kernel = 'linear')
# write results
y_fc3 = predict(model3, newdata = train_var, type = 'class')

# model evaluation
confusionMatrix(train_y_act, y_fc3)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8913043
```

3 Results

The models are trained against the training set. For the evaluation of results the validation data set is now used (only in this section for validation):

```
################################################################################+
# Prediction Model 1 Validation
################################################################################+

# predict responses with log function
y_fc1_glm = predict(model1, newdata = validate_var_nzv, type = 'response')

# translate into binary results of cancer
y_fc1 <- factor(ifelse(y_fc1_glm > 0.5, 1, 0))

# model evaluation
confusionMatrix(validate_y_act, y_fc1)
```

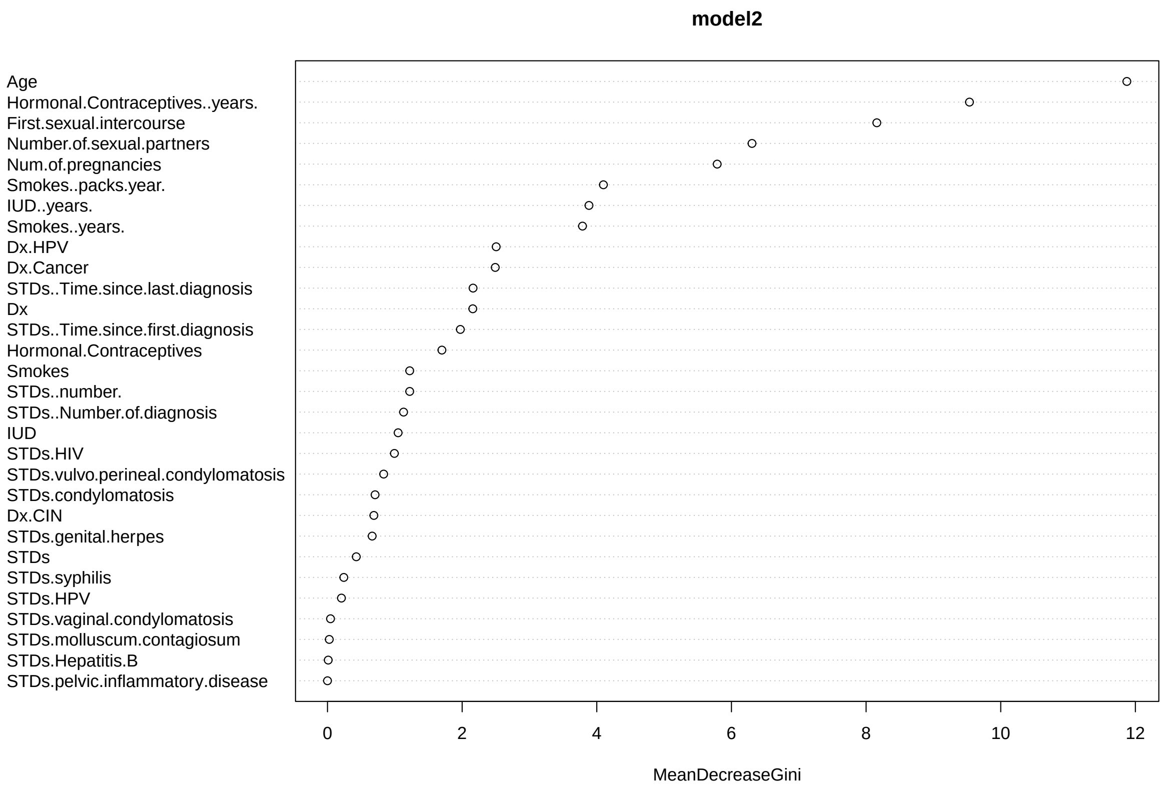
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 229   1
##          1  30   0
##
##                Accuracy : 0.8808
##                  95% CI : (0.835, 0.9175)
##     No Information Rate : 0.9962
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0075
##
##  Mcnemar's Test P-Value : 4.932e-07
##
##             Sensitivity : 0.8842
##             Specificity : 0.0000
##          Pos Pred Value : 0.9957
##          Neg Pred Value : 0.0000
##              Prevalence : 0.9962
##          Detection Rate : 0.8808
##    Detection Prevalence : 0.8846
##       Balanced Accuracy : 0.4421
##
##        'Positive' Class : 0
##
```

```
##################################################################################+
# Prediction Model 2 Validation
##################################################################################+

# write results
y_fc2 = predict(model2, newdata = validate_var, type = 'class')

# model evaluation
confusionMatrix(validate_y_act, y_fc2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 224   6
##          1  28   2
##
##                Accuracy : 0.8692
##                  95% CI : (0.8221, 0.9077)
##     No Information Rate : 0.9692
##     P-Value [Acc > NIR] : 1.0000000
##
##                   Kappa : 0.0596
##
##  Mcnemar's Test P-Value : 0.0003164
##
##             Sensitivity : 0.88889
##             Specificity : 0.25000
##          Pos Pred Value : 0.97391
##          Neg Pred Value : 0.06667
##              Prevalence : 0.96923
##          Detection Rate : 0.86154
##    Detection Prevalence : 0.88462
##       Balanced Accuracy : 0.56944
##
##        'Positive' Class : 0
##
```



```r
###############################################################################+
# Prediction Model 3 Validation
###############################################################################+

# write results
y_fc3 = predict(model3, newdata = validate_var, type = 'class')

# model evaluation
confusionMatrix(validate_y_act, y_fc3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 226   4
##          1  28   2
##
##                Accuracy : 0.8769
##                  95% CI : (0.8307, 0.9143)
##     No Information Rate : 0.9769
##     P-Value [Acc > NIR] : 1
##
```

```
##                  Kappa : 0.0756
##
##  Mcnemar's Test P-Value : 4.785e-05
##
##             Sensitivity : 0.88976
##             Specificity : 0.33333
##          Pos Pred Value : 0.98261
##          Neg Pred Value : 0.06667
##              Prevalence : 0.97692
##          Detection Rate : 0.86923
##    Detection Prevalence : 0.88462
##       Balanced Accuracy : 0.61155
##
##        'Positive' Class : 0
##
```

All models can retain their overall good accuracy of >.86. Overall accuracy accounts for True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN).

The Sensitivity reflects how well actual positives are recognized: (TP)/(TP+FN). All three models show a high Sensitivity of >.87 and therefore are good in predicting positive cases. But they differ when looking at the accuracy of actual negatives being classified correctly. This is expressed as Specificity = (TN)/(TN+FP).

The Specificity of 0 for the Logistical Regression shows this model is not capable to detect any actual negative value correctly. The parametrization either needs rework or data preparation for this needs to be adopted. Specificity of the Random Forrest was higher with .25 and SVM had the best result with .33. This means from this peer group the SVM model was te best to also predict actual negative cases correctly, but results are still imbalanced and not on level with the high sensitivity an overall accuracy.

4 Conclusion

The validation prediction shows overall good accuracy. The SVM had the best Specificity & Sensitivity pair, but needs further fine tuning to improve the true negative detection. Random Forrests showed also a imbalanced pair of true positive & negative detection, but can be validated with a low reliability. The Logistical regression converged, but with elimination of the near zero values subset any Specificity could be established and therefore the model is not validated in the current form.

The analysis already revealed further improvement approaches. First, the highly correlated variables need to be further aggregated into single dimensions to reduce noise and complexity for the models. Secondly, the near zero variation analysis showed many variables with a low number of unique values. The impact of those could also the explanation for missing information that is needed to form a valid prediction. Lastly, other models like k-Means or Neural Nets could lift better classifications, as unsupervised classification models are known to work well on such classification problems.