CYO Project - Hospital death

Ronaldo Benjamin Montes Kohler

25/2/2022

INTRODUCTION :

Machine learning is a big challenge it requires a lot mathematics,skills, time and patience, in this project we are going to analyze and try to predict a hospital death based on different predictors and different algorithms (glm , knn , random forest , rpart), the principal objective is to reduce the Rmse (an stadistic formula that measures the amount of error between two sets of data), the data comes from a page called kaggle and the creator and owner of the information you can find it in the following link .

https://www.kaggle.com/mitishaagarwal/patient.

Loading and dowloading all the packages from R that we are going to use in this project.

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(rpart.plot)) install.packages("rpart.plot", repos = "http://cran.us.r-project.org")
if(!require(ranger)) install.packages("ranger", repos = "http://cran.us.r-project.org")
if(!require(class)) install.packages("class", repos = "http://cran.us.r-project.org")
if(!require(gam)) install.packages("gam", repos = "http://cran.us.r-project.org")
if(!require(splines)) install.packages("splines", repos = "http://cran.us.r-project.org")
if(!require(foreach)) install.packages("foreach", repos = "http://cran.us.r-project.org")
if(!require(MASS)) install.packages("MASS", repos = "http://cran.us.r-project.org")
if(!require(C50)) install.packages("C50", repos = "http://cran.us.r-project.org")


library(tidyverse)
library(caret)
library(data.table)
```

```
library(lubridate)
library(randomForest)
library(dslabs)
library(rpart)
library(rpart.plot)
library(ranger)
library(class)
library(MASS)
library(gam)
library(splines)
library(foreach)
library(C50)
```

Methods/analysis :

Downloading the csv data and reading into a data frame.

```
Survival_Prediction <-read.csv('https://www.dropbox.com/s/angh53kkzc4awo6/dataset.csv?dl=1')
```

Now first we look at the data from the package with the function head(), it will illustrate what it contains the columns names and the type of data inside.

```
head(Survival_Prediction)
```

```
##   encounter_id patient_id hospital_id age   bmi elective_surgery ethnicity
## 1        66154      25312         118  68 22.73                0 Caucasian
## 2       114252      59342          81  77 27.42                0 Caucasian
## 3       119783      50777         118  25 31.95                0 Caucasian
## 4        79267      46918         118  81 22.64                1 Caucasian
## 5        92056      34377          33  19    NA                0 Caucasian
## 6        33181      74489          83  67 27.56                0 Caucasian
##   gender height          icu_admit_source icu_id icu_stay_type     icu_type
## 1      M  180.3                     Floor     92         admit        CTICU
## 2      F  160.0                     Floor     90         admit Med-Surg ICU
## 3      F  172.7      Accident & Emergency     93         admit Med-Surg ICU
## 4      F  165.1 Operating Room / Recovery     92         admit        CTICU
## 5      M  188.0      Accident & Emergency     91         admit Med-Surg ICU
## 6      M  190.5      Accident & Emergency     95         admit Med-Surg ICU
##   pre_icu_los_days weight apache_2_diagnosis apache_3j_diagnosis
## 1      0.541666667   73.9                113              502.01
## 2      0.927777778   70.2                108              203.01
## 3      0.000694444   95.3                122              703.03
## 4      0.000694444   61.7                203             1206.03
## 5      0.073611111     NA                119              601.01
```

```
## 6      0.000694444  100.0               301             403.01
##   apache_post_operative arf_apache gcs_eyes_apache gcs_motor_apache
## 1                     0          0               3                6
## 2                     0          0               1                3
## 3                     0          0               3                6
## 4                     1          0               4                6
## 5                     0          0              NA               NA
## 6                     0          0               4                6
##   gcs_unable_apache gcs_verbal_apache heart_rate_apache intubated_apache
## 1                 0                 4               118                0
## 2                 0                 1               120                0
## 3                 0                 5               102                0
## 4                 0                 5               114                1
## 5                NA                NA                60                0
## 6                 0                 5               113                0
##   map_apache resprate_apache temp_apache ventilated_apache d1_diasbp_max
## 1         40              36        39.3                 0            68
## 2         46              33        35.1                 1            95
## 3         68              37        36.7                 0            88
## 4         60               4        34.8                 1            48
## 5        103              16        36.7                 0            99
## 6        130              35        36.6                 0           100
##   d1_diasbp_min d1_diasbp_noninvasive_max d1_diasbp_noninvasive_min
## 1            37                        68                       37
## 2            31                        95                       31
## 3            48                        88                       48
## 4            42                        48                       42
## 5            57                        99                       57
## 6            61                       100                       61
##   d1_heartrate_max d1_heartrate_min d1_mbp_max d1_mbp_min
## 1              119              72         89         46
## 2              118              72        120         38
## 3               96              68        102         68
## 4              116              92         84         84
## 5               89              60        104         90
## 6              113              83        127         80
##   d1_mbp_noninvasive_max d1_mbp_noninvasive_min d1_resprate_max d1_resprate_min
## 1                     89                     46              34              10
## 2                    120                     38              32              12
## 3                    102                     68              21               8
## 4                     84                     84              23               7
## 5                    104                     90              18              16
## 6                    127                     80              32              10
##   d1_spo2_max d1_spo2_min d1_sysbp_max d1_sysbp_min d1_sysbp_noninvasive_max
## 1         100          74          131           73                      131
## 2         100          70          159           67                      159
## 3          98          91          148          105                      148
## 4         100          95          158           84                      158
## 5         100          96          147          120                      147
## 6          97          91          173          107                      173
##   d1_sysbp_noninvasive_min d1_temp_max d1_temp_min h1_diasbp_max h1_diasbp_min
## 1                       73        39.9        37.2            68            63
## 2                       67        36.3        35.1            61            48
## 3                      105        37.0        36.7            88            58
```

```
## 4                      84        38.0        34.8            62            44
## 5                     120        37.2        36.7            99            68
## 6                     107        36.8        36.6            89            89
##   h1_diasbp_noninvasive_max h1_diasbp_noninvasive_min h1_heartrate_max
## 1                        68                        63              119
## 2                        61                        48              114
## 3                        88                        58               96
## 4                        NA                        NA              100
## 5                        99                        68               89
## 6                        89                        89               83
##   h1_heartrate_min h1_mbp_max h1_mbp_min h1_mbp_noninvasive_max
## 1              108         86         85                     86
## 2              100         85         57                     85
## 3               78         91         83                     91
## 4               96         92         71                     NA
## 5               76        104         92                    104
## 6               83        111        111                    111
##   h1_mbp_noninvasive_min h1_resprate_max h1_resprate_min h1_spo2_max
## 1                     85              26              18         100
## 2                     57              31              28          95
## 3                     83              20              16          98
## 4                     NA              12              11         100
## 5                     92              NA              NA         100
## 6                    111              12              12          97
##   h1_spo2_min h1_sysbp_max h1_sysbp_min h1_sysbp_noninvasive_max
## 1          74          131          115                     131
## 2          70           95           71                      95
## 3          91          148          124                     148
## 4          99          136          106                      NA
## 5         100          130          120                     130
## 6          97          143          143                     143
##   h1_sysbp_noninvasive_min d1_glucose_max d1_glucose_min d1_potassium_max
## 1                      115            168            109              4.0
## 2                       71            145            128              4.2
## 3                      124             NA             NA               NA
## 4                       NA            185             88              5.0
## 5                      120             NA             NA               NA
## 6                      143            156            125              3.9
##   d1_potassium_min apache_4a_hospital_death_prob apache_4a_icu_death_prob aids
## 1              3.4                          0.10                     0.05    0
## 2              3.8                          0.47                     0.29    0
## 3               NA                          0.00                     0.00    0
## 4              3.5                          0.04                     0.03    0
## 5               NA                            NA                       NA    0
## 6              3.7                          0.05                     0.02    0
##   cirrhosis diabetes_mellitus hepatic_failure immunosuppression leukemia
## 1         0                 1               0                  0        0
## 2         0                 1               0                  0        0
## 3         0                 0               0                  0        0
## 4         0                 0               0                  0        0
## 5         0                 0               0                  0        0
## 6         0                 1               0                  0        0
##   lymphoma solid_tumor_with_metastasis apache_3j_bodysystem apache_2_bodysystem
## 1        0                          0              Sepsis      Cardiovascular
```

```
## 2        0                          0          Respiratory         Respiratory
## 3        0                          0            Metabolic           Metabolic
## 4        0                          0       Cardiovascular      Cardiovascular
## 5        0                          0              Trauma              Trauma
## 6        0                          0         Neurological          Neurologic
##    X hospital_death
## 1 NA              0
## 2 NA              0
## 3 NA              0
## 4 NA              0
## 5 NA              0
## 6 NA              0
```

We notice that there is a column called X in the data frame which most of the data is NA, we clean the data from any NA value.

```
Survival_Prediction  <- Survival_Prediction[,-84]


Survival_Prediction <- Survival_Prediction %>%
  drop_na()

sum(is.na(Survival_Prediction$age))
```

```
## [1] 0
```

We also notice that there is a lot of specificity in the height , weight and the temperature, It will not be very helpful to have the data like this since when grouping them there would be many groups of a single value and the analysis would be much more difficult, we round the values to the nearest integer.

```
Survival_Prediction$height <- round(Survival_Prediction$height)

Survival_Prediction$weight <- round(Survival_Prediction$weight)

Survival_Prediction$temp_apache <- round(Survival_Prediction$temp_apache)

Survival_Prediction$d1_temp_min   <- round(Survival_Prediction$d1_temp_min  )

Survival_Prediction$d1_temp_max    <- round(Survival_Prediction$d1_temp_max   )
```

Due to data visualization problems we have decided to create a special variable.

```
Survival_Prediction_vi <- Survival_Prediction
```

Some columns has binary data or specific groups, so we decide to convert it to a factor variable.

```
Survival_Prediction$hospital_death <- as.factor(Survival_Prediction$hospital_death)
Survival_Prediction$gcs_eyes_apache <- as.factor(Survival_Prediction$gcs_eyes_apache)
Survival_Prediction$gcs_motor_apache <- as.factor(Survival_Prediction$gcs_motor_apache)
Survival_Prediction$gcs_verbal_apache <- as.factor(Survival_Prediction$gcs_verbal_apache)
Survival_Prediction$intubated_apache <- as.factor(Survival_Prediction$intubated_apache)
Survival_Prediction$aids <- as.factor(Survival_Prediction$aids)
Survival_Prediction$cirrhosis <- as.factor(Survival_Prediction$cirrhosis)
Survival_Prediction$solid_tumor_with_metastasis <- as.factor(Survival_Prediction$solid_tumor_with_metas
```

We are going to split the data in two parts one for training the algorithm (90%), and another to validate(10%) the results.

```
test_index <- createDataPartition(y = Survival_Prediction$hospital_death, times = 1, p = 0.1, list = FA
edx <- Survival_Prediction[-test_index,]
temp <- Survival_Prediction[test_index,]

validation <- temp %>%
  semi_join(edx, by = "hospital_death")

removed <- anti_join(temp, validation)
```

```
## Joining, by = c("encounter_id", "patient_id", "hospital_id", "age", "bmi", "elective_surgery", "ethn
```

```
edx <- rbind(edx, removed)
```

We are only going to use the EDX set to realize all the machine learning work, so the EDX data set is going to be splitted on a train set(90%) and a test set(10%).

```
test_index <- createDataPartition(y = edx$hospital_death, times = 1, p = 0.1, list = FALSE)
edx_train <- edx[-test_index,]
temp <- edx[test_index,]
```

```
edx_test <- temp %>%
  semi_join(edx, by = "hospital_death")

removed <- anti_join(temp, edx_test)
```

```
## Joining, by = c("encounter_id", "patient_id", "hospital_id", "age", "bmi", "elective_surgery", "ethni
```

```
edx_train <- rbind(edx_train, removed)
```

The data has 84 columns, trying to explore all of theses columns might take a lot time, so we are only going to use a few columns more flashy.

```
ncol(edx_train)
```

```
## [1] 84
```

Some columns has the label Apache that is an acronym for Acute Physiology and Chronic Health Evaluation, It is applied within 24 hours of admission of a patient to an intensive care unit (ICU).

```
colnames(edx_train)
```

```
##  [1] "encounter_id"                "patient_id"
##  [3] "hospital_id"                 "age"
##  [5] "bmi"                         "elective_surgery"
##  [7] "ethnicity"                   "gender"
##  [9] "height"                      "icu_admit_source"
## [11] "icu_id"                      "icu_stay_type"
## [13] "icu_type"                    "pre_icu_los_days"
## [15] "weight"                      "apache_2_diagnosis"
## [17] "apache_3j_diagnosis"         "apache_post_operative"
## [19] "arf_apache"                  "gcs_eyes_apache"
## [21] "gcs_motor_apache"            "gcs_unable_apache"
## [23] "gcs_verbal_apache"           "heart_rate_apache"
## [25] "intubated_apache"            "map_apache"
## [27] "resprate_apache"             "temp_apache"
## [29] "ventilated_apache"           "d1_diasbp_max"
## [31] "d1_diasbp_min"               "d1_diasbp_noninvasive_max"
## [33] "d1_diasbp_noninvasive_min"   "d1_heartrate_max"
## [35] "d1_heartrate_min"            "d1_mbp_max"
## [37] "d1_mbp_min"                  "d1_mbp_noninvasive_max"
## [39] "d1_mbp_noninvasive_min"      "d1_resprate_max"
## [41] "d1_resprate_min"             "d1_spo2_max"
```

```
## [43] "d1_spo2_min"                   "d1_sysbp_max"
## [45] "d1_sysbp_min"                  "d1_sysbp_noninvasive_max"
## [47] "d1_sysbp_noninvasive_min"      "d1_temp_max"
## [49] "d1_temp_min"                   "h1_diasbp_max"
## [51] "h1_diasbp_min"                 "h1_diasbp_noninvasive_max"
## [53] "h1_diasbp_noninvasive_min"     "h1_heartrate_max"
## [55] "h1_heartrate_min"              "h1_mbp_max"
## [57] "h1_mbp_min"                    "h1_mbp_noninvasive_max"
## [59] "h1_mbp_noninvasive_min"        "h1_resprate_max"
## [61] "h1_resprate_min"               "h1_spo2_max"
## [63] "h1_spo2_min"                   "h1_sysbp_max"
## [65] "h1_sysbp_min"                  "h1_sysbp_noninvasive_max"
## [67] "h1_sysbp_noninvasive_min"      "d1_glucose_max"
## [69] "d1_glucose_min"                "d1_potassium_max"
## [71] "d1_potassium_min"              "apache_4a_hospital_death_prob"
## [73] "apache_4a_icu_death_prob"      "aids"
## [75] "cirrhosis"                     "diabetes_mellitus"
## [77] "hepatic_failure"               "immunosuppression"
## [79] "leukemia"                      "lymphoma"
## [81] "solid_tumor_with_metastasis"   "apache_3j_bodysystem"
## [83] "apache_2_bodysystem"           "hospital_death"
```

In this project we are going to use some machine learning algorithms, to analyze and use the algorithms we need to understand how they work.

Random forest is a supervised machine learning algorithm that uses ensemble learning method for regression, 'the idea of random forests is to generate many predictors, each using regression or classification trees, and then forming a final prediction based on the average prediction of all these trees'.

C5.0 model works by splitting the sample based on the field that provides the maximum information gain. Each sub-sample defined by the first split is then split again, usually based on a different field, and the process repeats until the sub samples cannot be split any further. Finally, the lowest-level splits are reexamined.

Generalized Additive Model using LOESS

The additive model generalizes the linear model by modeling the expected value of Y as

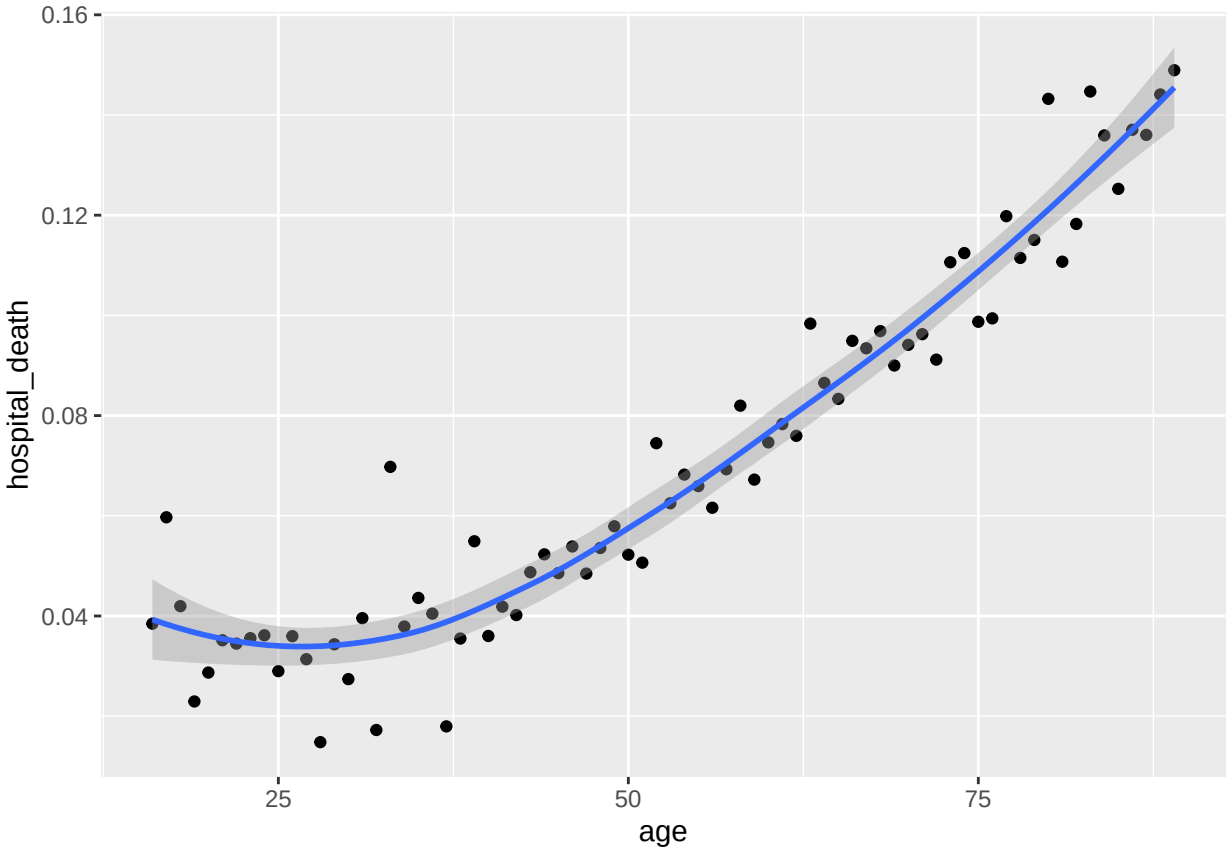
$$E(Y) = f(X1 \ldots.. Xp) = s0 + s1(X1) + \ldots\ldots + sp(Xp)$$

Where si(X), i = 1 ..... p , are smooth functions. These functions are estimated in a nonparametric fashion.

Exploring the data the first intuition that we had was that the probability of death of a older person was greater than a young, so we plot the mean hospital death grouped by the age.

```
Survival_Prediction_vi %>%
  group_by(age) %>%
  summarize(hospital_death = mean(hospital_death), age =  mean(age)) %>%
  ggplot(aes(age, hospital_death)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



There's a strong evidence that an older person has a higher of die than a younger one.

RandomForest() function allows us to run Random Forest Regression.

Testing the age on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age, data = edx_train)

y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.914159
```

```
sum(y_hat_rf == 1)
```

```
## [1] 0
```

Train() function with method = 'C5.0Tree' parameter allows us to run C5.0Tree algorithm.

Testing the first model C5.0Tree.

```
fit_ct <- train(hospital_death ~ age, data = edx_train, method = 'C5.0Tree' )

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.914159
```

```
sum(y_hat_ct == 1)
```

```
## [1] 0
```

Train() function with method = 'gamLoess' parameter allows us to run gamLoess algorithm.

Testing the first model gamLoess.

```
fit_gam <- train(hospital_death ~ age, data = edx_train, method = 'gamLoess' )

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

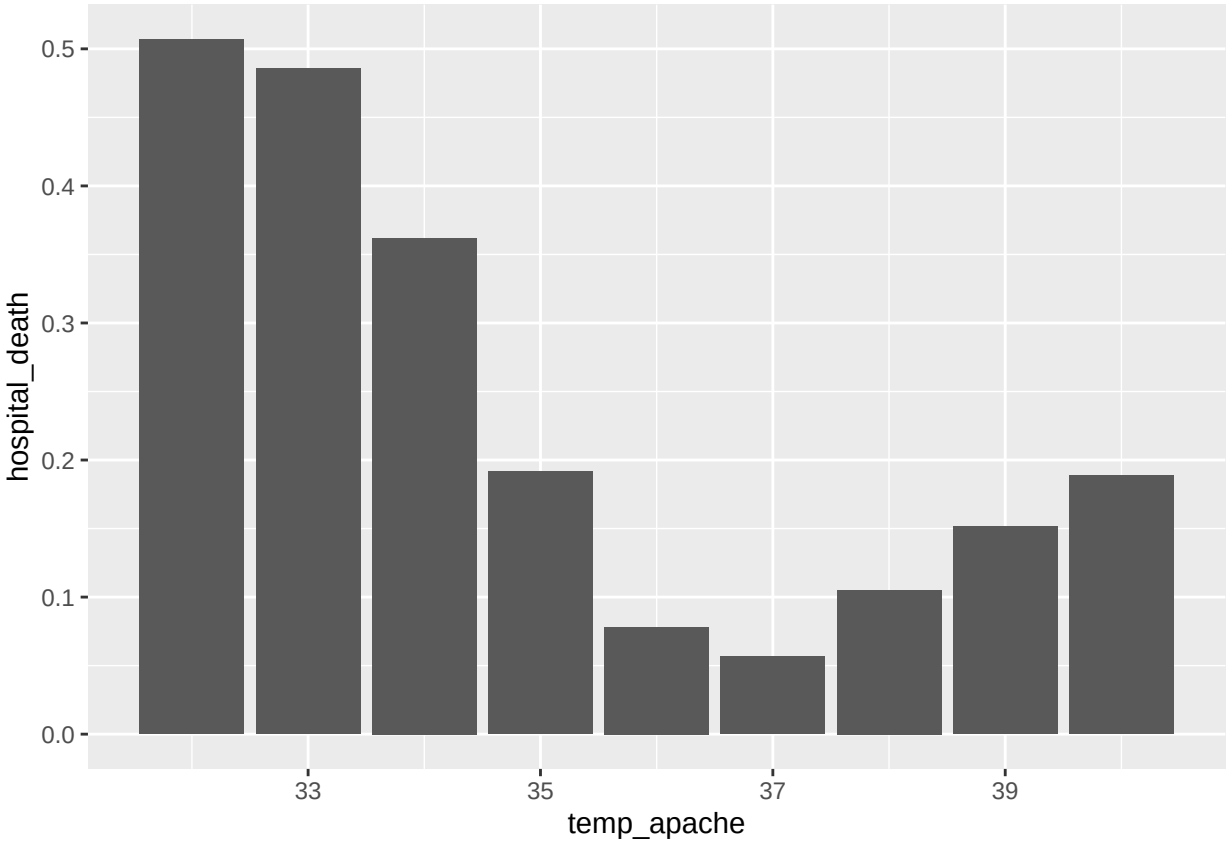
```
## [1] 0.914159
```

```
sum(y_hat_gam == 1)
```

```
## [1] 0
```

The person's temperature can be a great indication of risk of death especially when entering intensive care.

```
Survival_Prediction_vi %>%
  group_by(temp_apache ) %>%
  summarize(temp_apache  = mean(temp_apache , na.rm = TRUE),hospital_death = mean(hospital_death , na.r
  ggplot(aes(temp_apache , hospital_death )) +
  geom_col()
```



Testing the temp_apache on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + temp_apache , data = edx_train)


y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.913966
```

```
sum(y_hat_rf == 1)
```

```
## [1] 31
```

Testing the temp_apache on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age + temp_apache , data = edx_train , method = 'C5.0Tree')
```

```
y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.914159
```

```
sum(y_hat_ct == 1)
```

```
## [1] 0
```

Testing the temp_apache on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache ,  data = edx_train, method = 'gamLoess' )

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

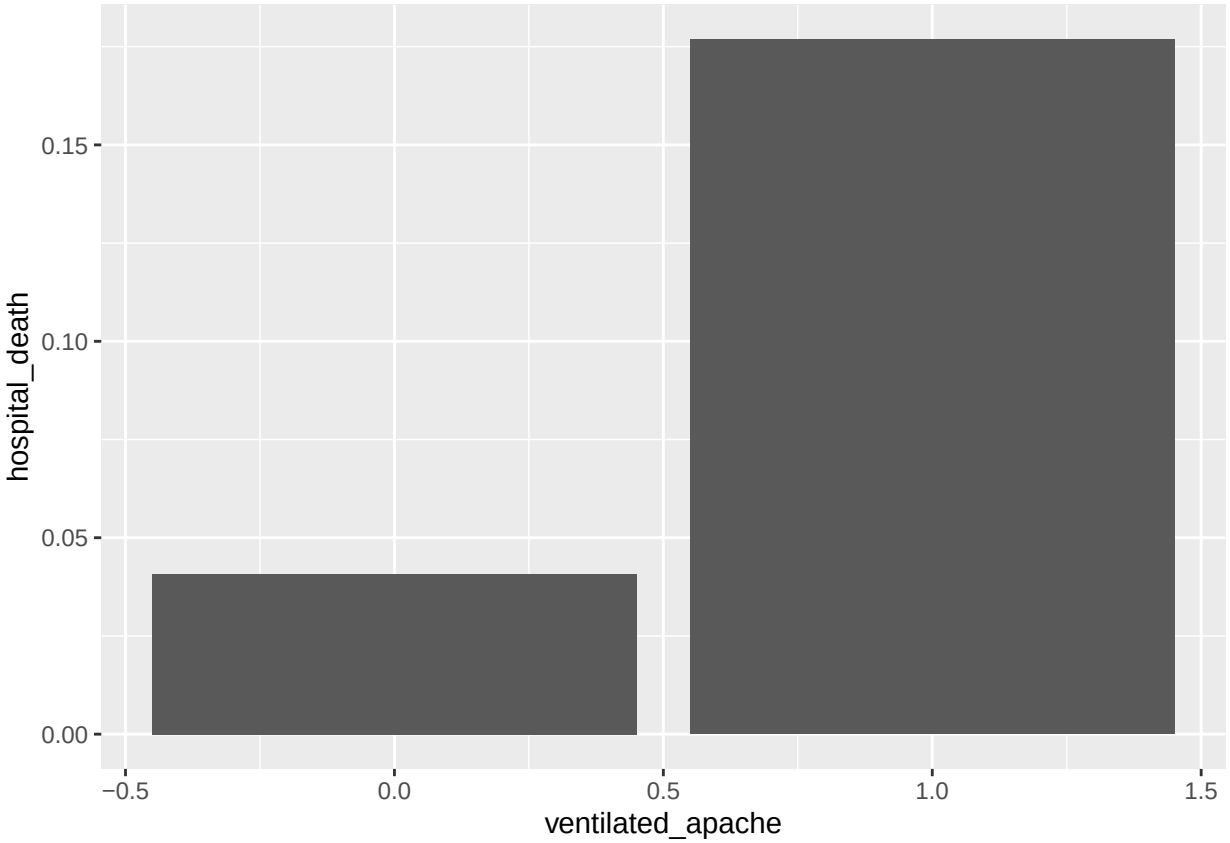
```
## [1] 0.9151235
```

```
sum(y_hat_gam == 1)
```

```
## [1] 7
```

Our intuition says that people who enter the mechanical ventilation area are more likely to die.

```
Survival_Prediction_vi %>%
  group_by(ventilated_apache ) %>%
  summarize(ventilated_apache  = mean(ventilated_apache , na.rm = TRUE),hospital_death = mean(hospital_
  ggplot(aes(ventilated_apache ,hospital_death  )) +
  geom_col()
```



The data is binary, where 1 one is this with mechanical ventilation and 0 is the opposite Analyzing the graph, we realize that people who actually enter mechanical ventilation have a greater chance of dying.

Testing the ventilated_apache on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age  + ventilated_apache + temp_apache , data = edx_train)


y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9149306
```

```
sum(y_hat_rf == 1)
```

```
## [1] 30
```

Testing the ventilated_apache on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age  + ventilated_apache + temp_apache , data = edx_train, method = 'C5

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.914159
```

```
sum(y_hat_ct == 1)
```

```
## [1] 0
```

Testing the ventilated_apache on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache        ,  data = edx_train, me

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9153164
```

```
sum(y_hat_gam == 1)
```

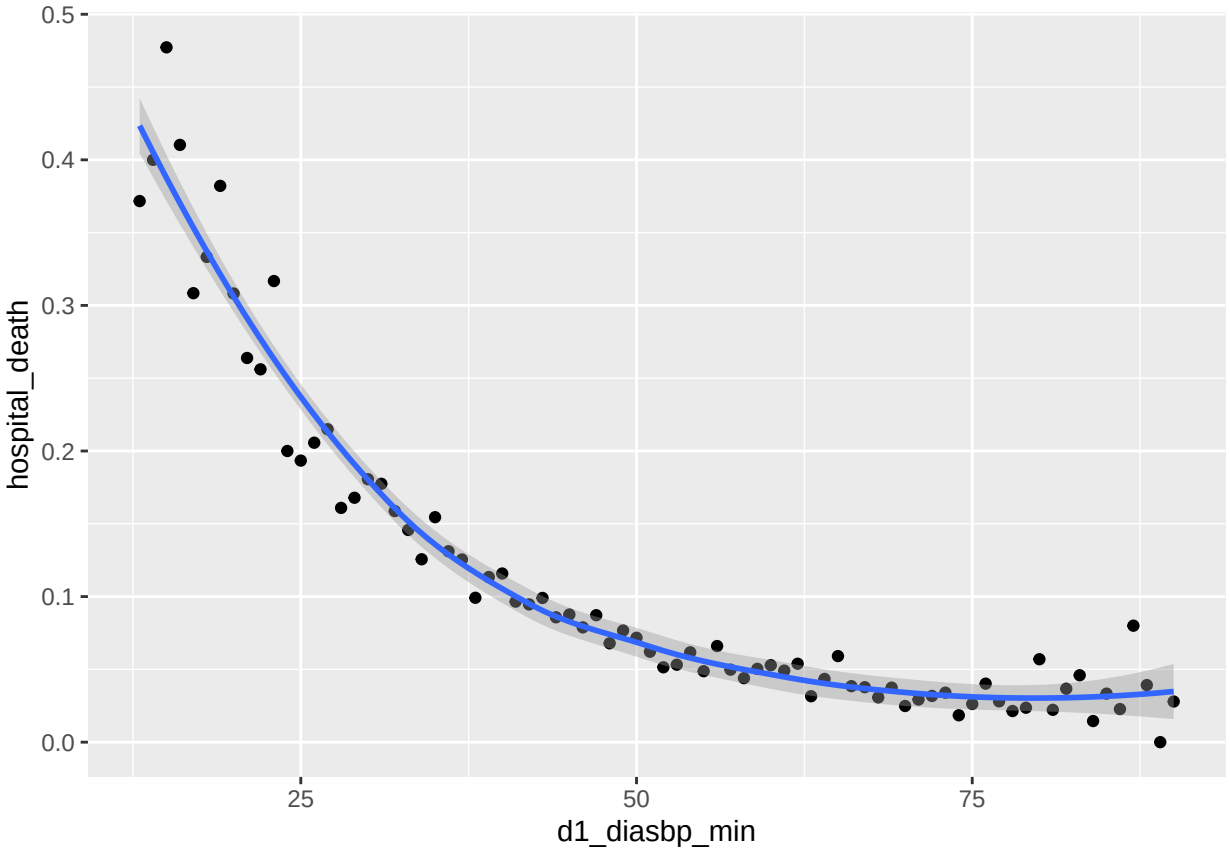
```
## [1] 12
```

Diasbp is an acronym of Diastolic blood pressure.

Can the diastolic blood pressure have an effect on the death of a person?

```
Survival_Prediction_vi %>%
  group_by(d1_diasbp_min   ) %>%
  summarize(d1_diasbp_min    = mean(d1_diasbp_min   , na.rm = TRUE),hospital_death = mean(hospital_deatl
  ggplot(aes(d1_diasbp_min   ,hospital_death  )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



There is a very clear relationship where having a lower pressure means more chances of death.

Testing the d1_diasbp_min on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min    , data
```

```
y_hat_rf <- predict(fit_rf, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9143519
```

```
sum(y_hat_rf == 1)
```

```
## [1] 75
```

Testing the d1_diasbp_min on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min  , data = edx

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9151235
```

```
sum(y_hat_ct == 1)
```

```
## [1] 27
```

Testing the d1__diasbp__min on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min          ,  da

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9157022
```

```
sum(y_hat_gam == 1)
```

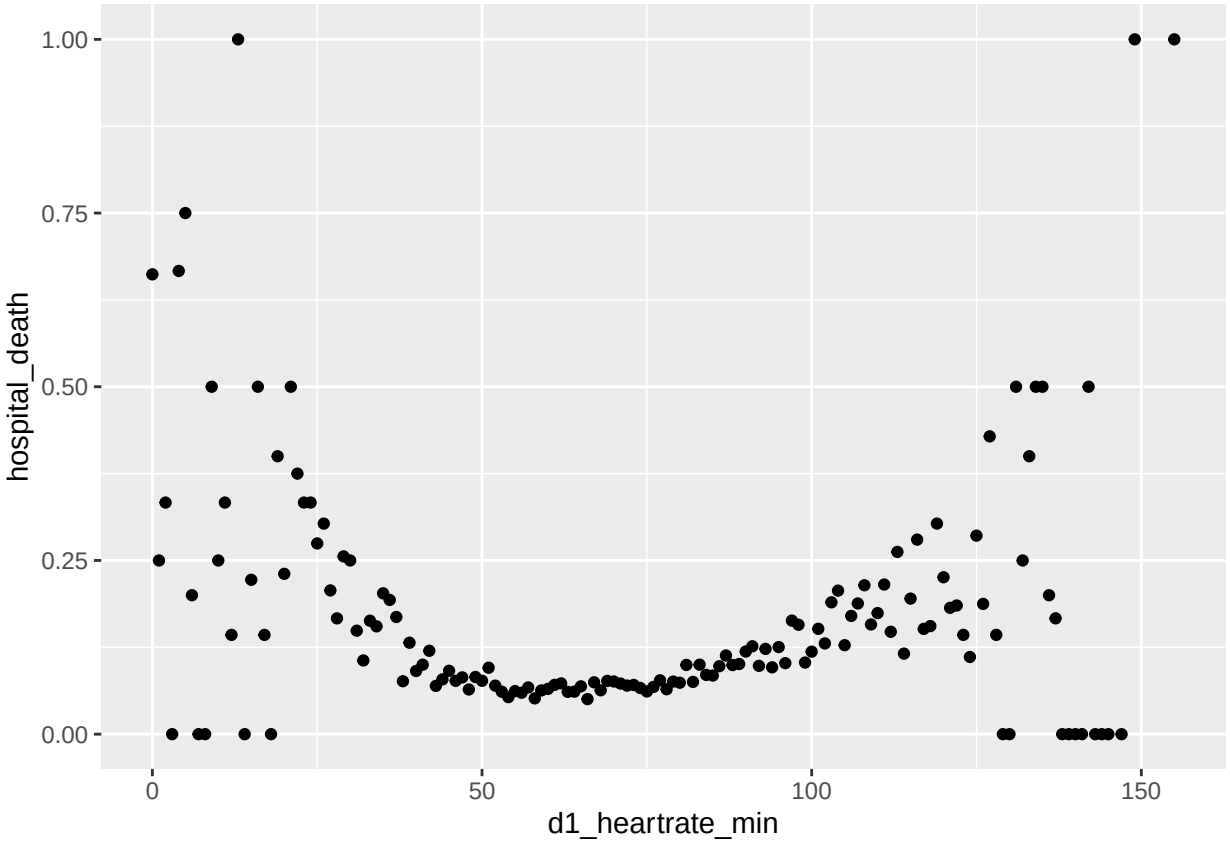
```
## [1] 42
```

Testing the min heart rate on the rpart algorithm.

The heart rate is always a determining factor in the death of a person.

```
Survival_Prediction_vi %>%
  group_by(d1_heartrate_min ) %>%
  summarize(d1_heartrate_min  = mean(d1_heartrate_min ),hospital_death = mean(hospital_death , na.rm = 
  ggplot(aes(d1_heartrate_min ,hospital_death , group =d1_heartrate_min   )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Testing the d1_heartrate_min on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear

y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9174383
```

```
sum(y_hat_rf == 1)
```

```
## [1] 101
```

Testing the d1_heartrate_min on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat

y_hat_ct <- predict(fit_ct, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9178241
```

```
sum(y_hat_ct == 1)
```

```
## [1] 45
```

Testing the d1__heartrate__min on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9184028
```

```
sum(y_hat_gam == 1)
```

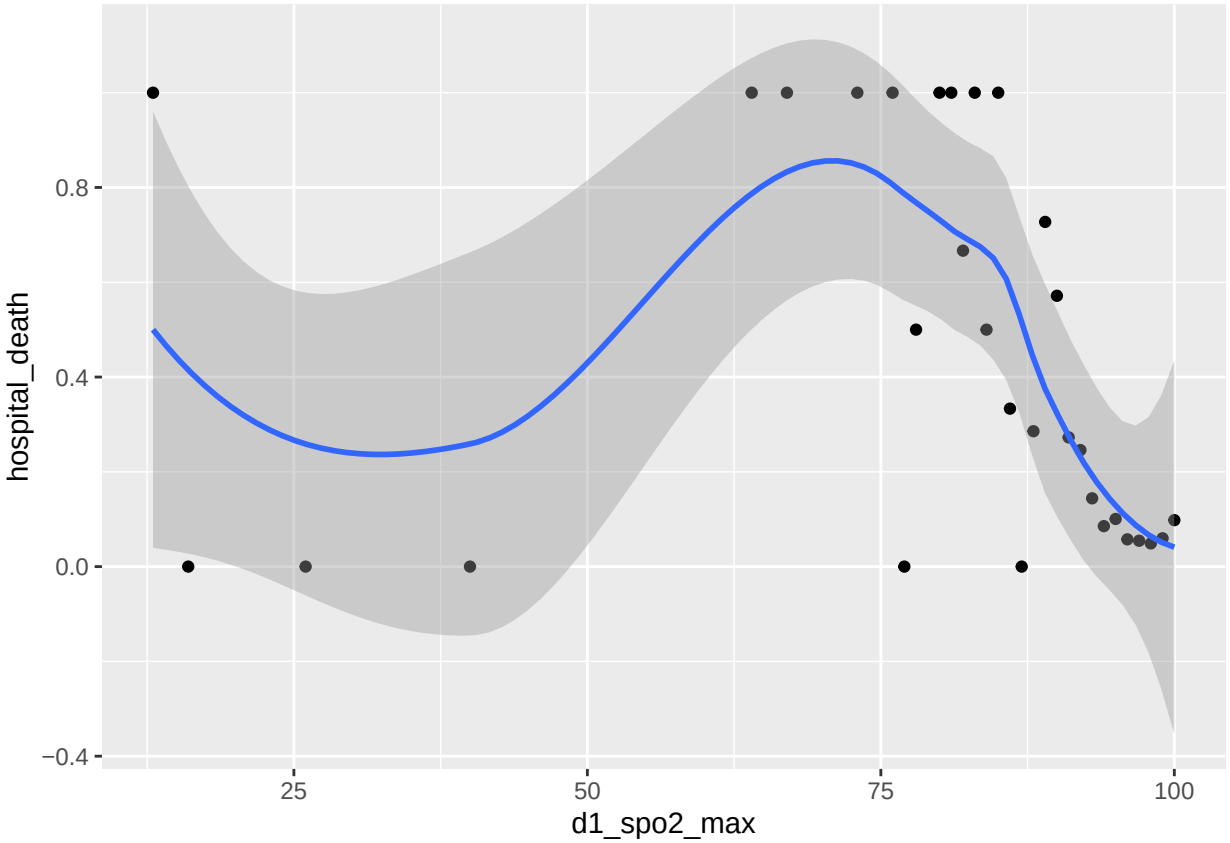
```
## [1] 66
```

SpO2 stands for Saturation of peripheral Oxygen, used to estimate the oxygen saturation of arterial blood .

Can max oxygen saturation of arterial blood have an impact on in-hospital death?

```
Survival_Prediction_vi %>%
  group_by(d1_spo2_max  ) %>%
  summarize(d1_spo2_max   = mean(d1_spo2_max  ),hospital_death = mean(hospital_death , na.rm = TRUE)) %
  ggplot(aes(d1_spo2_max  ,hospital_death  )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Testing the d1_spo2_max on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear
```

```
y_hat_rf <- predict(fit_rf, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9207176
```

```
sum(y_hat_rf == 1)
```

```
## [1] 94
```

Testing the d1_spo2_max on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat
```

```
y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9203318
```

```
sum(y_hat_ct == 1)
```

```
## [1] 70
```

Testing the d1_spo2_max on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9187886
```

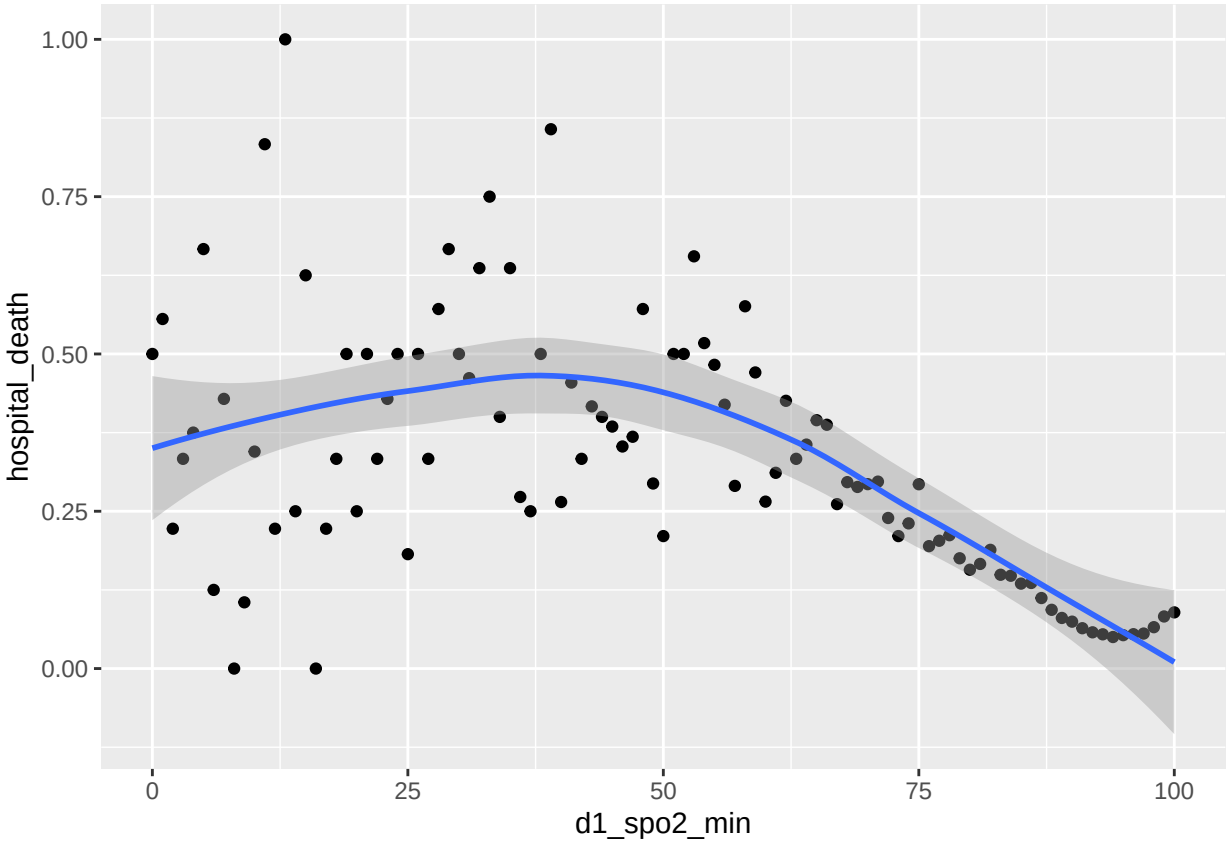
```
sum(y_hat_gam == 1)
```

```
## [1] 68
```

Can min oxygen saturation have an impact on in-hospital death?.

```
Survival_Prediction_vi %>%
  group_by(d1_spo2_min    ) %>%
  summarize(d1_spo2_min   = mean(d1_spo2_min  ),hospital_death = mean(hospital_death , na.rm = TRUE)) %
  ggplot(aes(d1_spo2_min  ,hospital_death  )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



There is a very strong relationship between the lower the oxygen saturation, the higher the chances of dying

Testing the d1_spo2_min on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear

y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9207176
```

```
sum(y_hat_rf == 1)
```

```
## [1] 100
```

Testing the d1_spo2_min on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat
```

```
y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9195602
```

```
sum(y_hat_ct == 1)
```

```
## [1] 76
```

Testing the d1_spo2_min on the gamLoess algorithm.m.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9214892
```

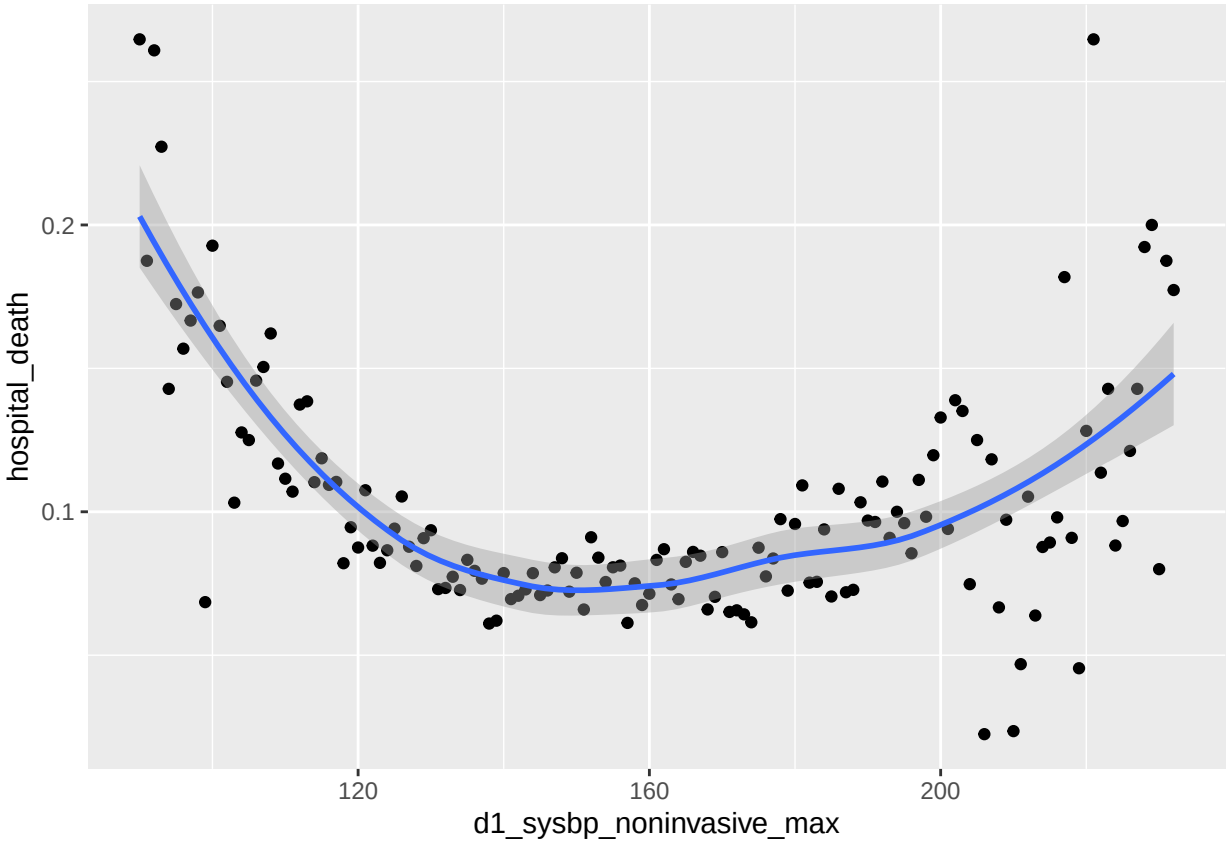
```
sum(y_hat_gam == 1)
```

```
## [1] 88
```

Sysbp or Systolic blood pressures, is the pressure exerted when the heart beats and blood is ejected into the arteries.

```
Survival_Prediction_vi %>%
  group_by(d1_sysbp_noninvasive_max  ) %>%
  summarize(d1_sysbp_noninvasive_max   = mean(d1_sysbp_noninvasive_max  ),hospital_death = mean(hospita
  ggplot(aes(d1_sysbp_noninvasive_max  ,hospital_death  )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Testing the d1_sysbp_noninvasive_max on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear
```

```
y_hat_rf <- predict(fit_rf, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9222608
```

```
sum(y_hat_rf == 1)
```

```
## [1] 98
```

Testing the d1_sysbp_noninvasive_max on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9191744
```

```
sum(y_hat_ct == 1)
```

```
## [1] 80
```

Testing the d1_sysbp_noninvasive_max on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

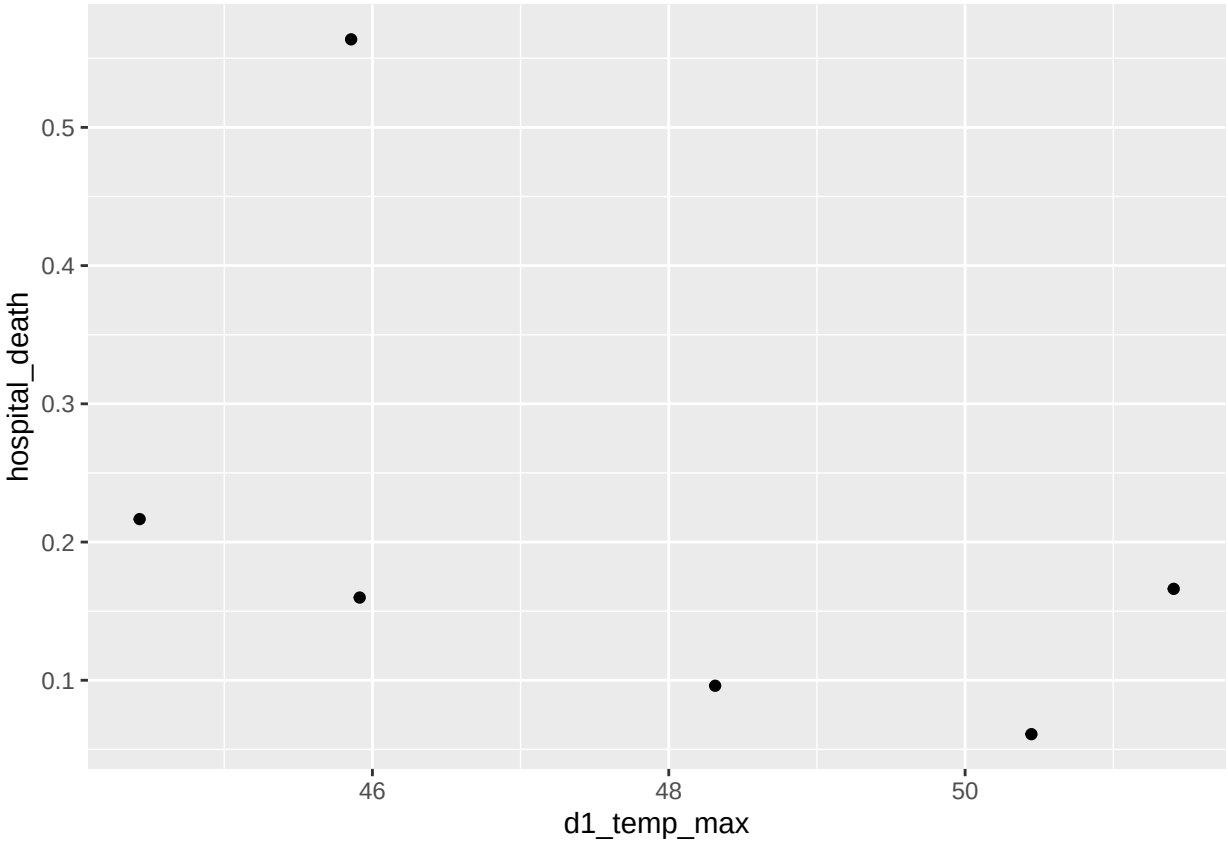
```
## [1] 0.9212963
```

```
sum(y_hat_gam == 1)
```

```
## [1] 93
```

The temperature of people is a great factor to analyze to detect diseases, could this parameter be fatal?

```
Survival_Prediction_vi %>%
  group_by(d1_temp_max ) %>%
  summarize(d1_temp_max  = mean(d1_diasbp_min),hospital_death = mean(hospital_death , na.rm = TRUE)) %>%
  ggplot(aes(d1_temp_max ,hospital_death  )) +
  geom_point()
```



Testing the d1_temp_max on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear
```

```
y_hat_rf <- predict(fit_rf, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9226466
```

```
sum(y_hat_rf == 1)
```

```
## [1] 118
```

Testing the d1_temp_max on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat
```

```
y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9224537
```

```
sum(y_hat_ct == 1)
```

```
## [1] 93
```

Testing the d1__temp__max on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9214892
```

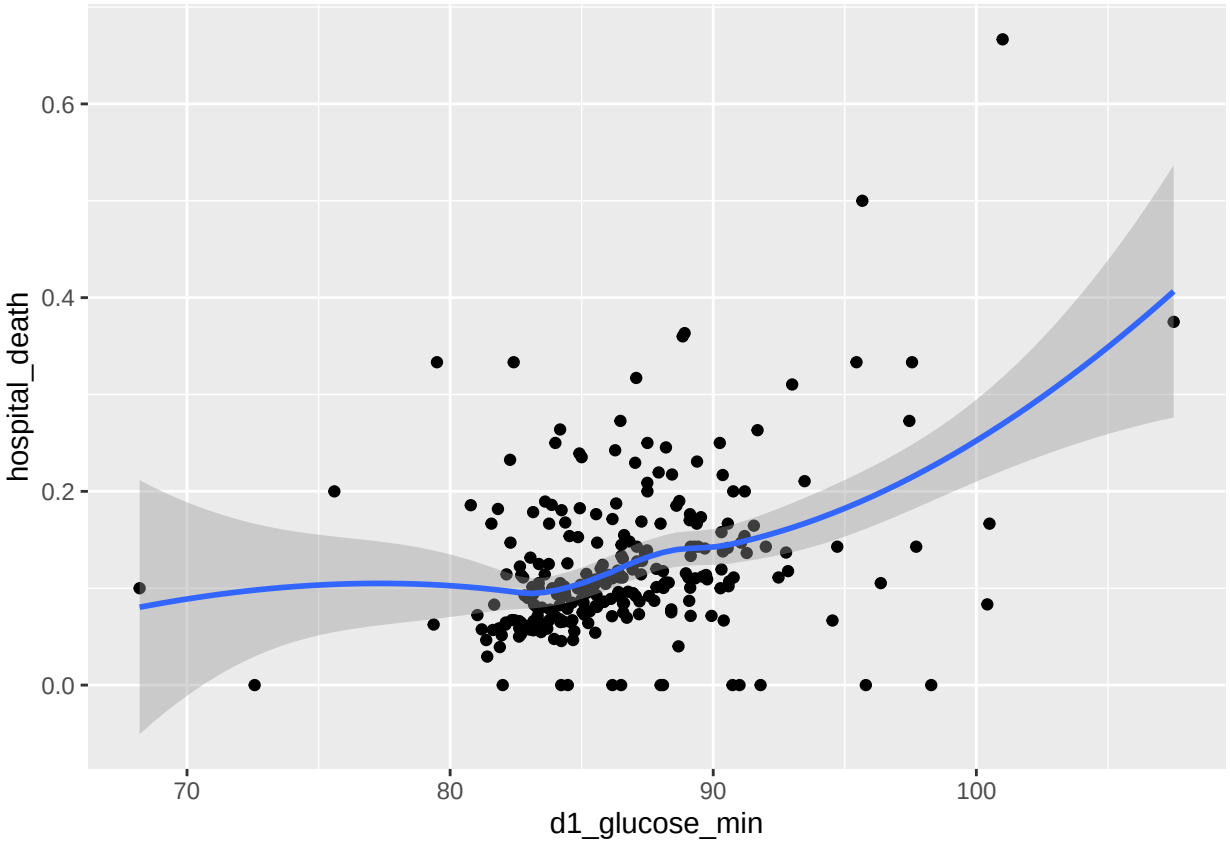
```
sum(y_hat_gam == 1)
```

```
## [1] 94
```

Glucose level is always a factor analyzed by doctors.

```
Survival_Prediction_vi %>%
  group_by(d1_glucose_min  ) %>%
  summarize(d1_glucose_min   = mean(h1_heartrate_min),hospital_death = mean(hospital_death , na.rm = TRU
  ggplot(aes(d1_glucose_min  ,hospital_death  )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



There seems to be some relationship between glucose level and hospital death.

Testing the d1__glucose__min on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear

y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9251543
```

```
sum(y_hat_rf == 1)
```

```
## [1] 111
```

Testing the d1__glucose__min on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9222608
```

```
sum(y_hat_ct == 1)
```

```
## [1] 116
```

Testing the d1__glucose__min on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

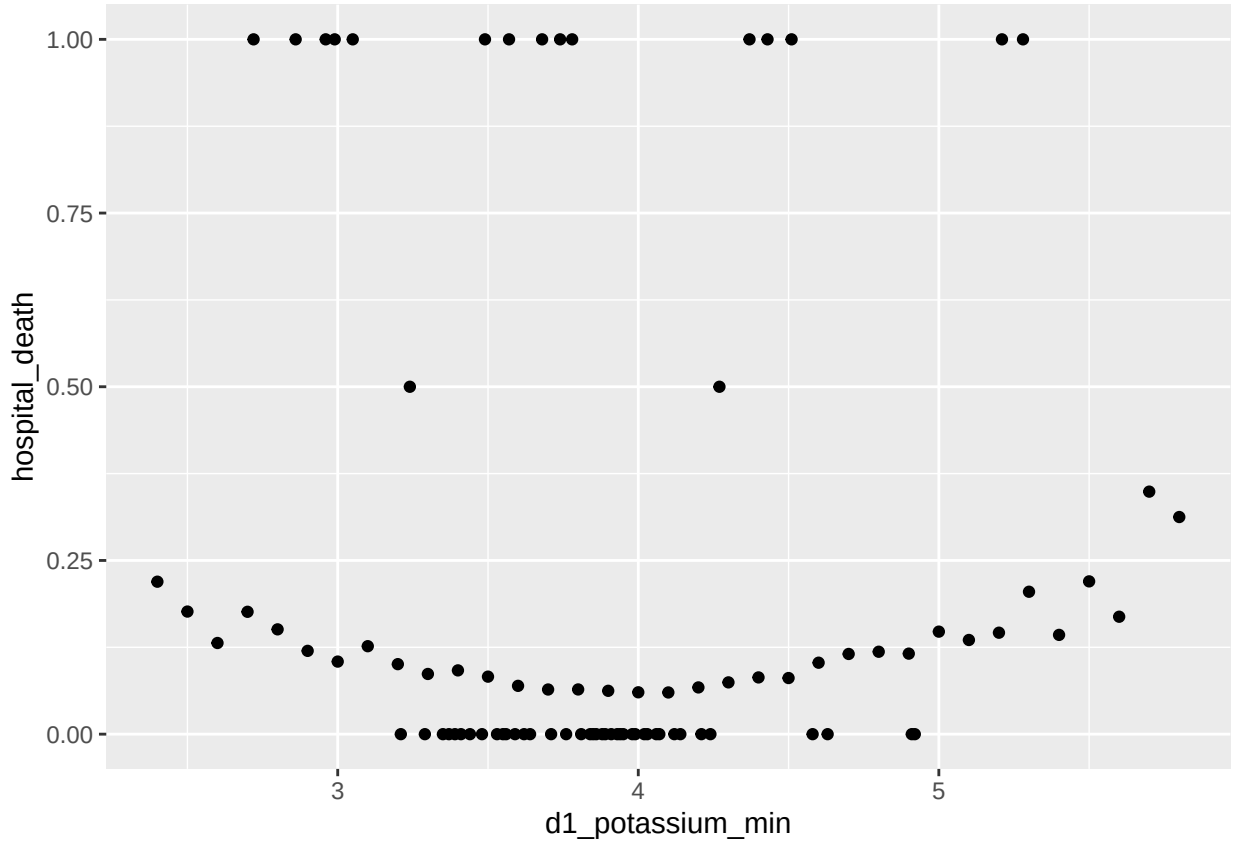
```
## [1] 0.9228395
```

```
sum(y_hat_gam == 1)
```

```
## [1] 97
```

Lack of minerals and poor diet can have an effect on people's health.

```
Survival_Prediction_vi %>%
  group_by(d1_potassium_min ) %>%
  summarize(d1_potassium_min  = mean(d1_potassium_min ),hospital_death = mean(hospital_death , na.rm = 
  ggplot(aes(d1_potassium_min ,hospital_death  )) +
  geom_point()
```



The points are scattered appearing to be unrelated.

Testing the d1_potassium_min on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear
```

```
y_hat_rf <- predict(fit_rf, edx_test)
```

```
mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9243827
```

```
sum(y_hat_rf == 1)
```

```
## [1] 107
```

Testing the d1_potassium_min on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat

y_hat_ct <- predict(fit_ct, edx_test)

mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9230324
```

```
sum(y_hat_ct == 1)
```

```
## [1] 116
```

Testing the d1_potassium_min on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra

y_hat_gam <- predict(fit_gam, edx_test)

mean(edx_test$hospital_death == y_hat_gam)
```

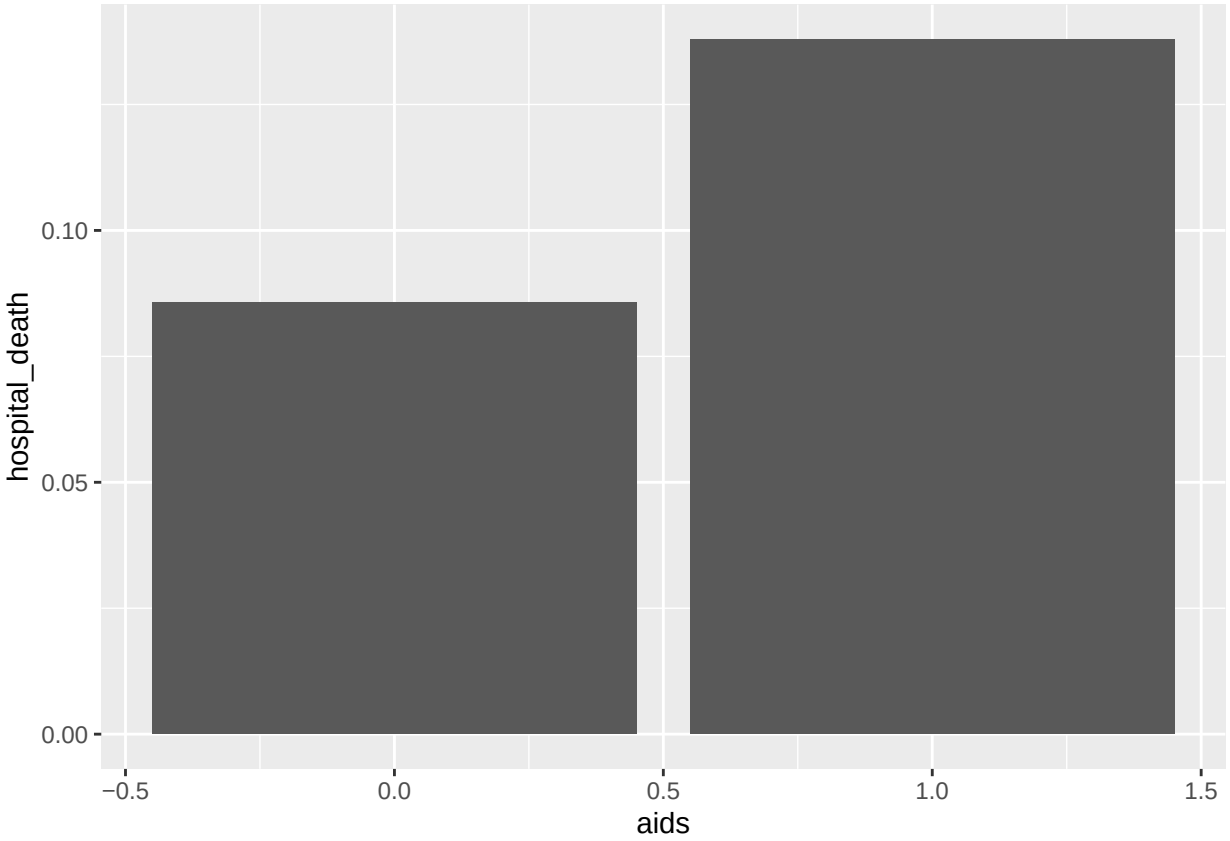
```
## [1] 0.9224537
```

```
sum(y_hat_gam == 1)
```

```
## [1] 101
```

Our intuition says that people who have AIDS are more likely to die. Is this true?

```
Survival_Prediction_vi %>%
  group_by(aids) %>%
  summarize(aids = mean(aids),hospital_death = mean(hospital_death , na.rm = TRUE)) %>%
  ggplot(aes(aids,hospital_death  )) +
  geom_col()
```



As we can see people who have AIDS have a little more chance of having a hospital death.

Testing the aids parameter on the random forest algorithm.

```
fit_rf <- randomForest(hospital_death ~ age + ventilated_apache + temp_apache + d1_diasbp_min + d1_hear

y_hat_rf <- predict(fit_rf, edx_test)

mean(edx_test$hospital_death == y_hat_rf)
```

```
## [1] 0.9232253
```

```
sum(y_hat_rf == 1)
```

```
## [1] 103
```

Testing the aids parameter on the C5.0Tree algorithm.

```
fit_ct <- train(hospital_death ~ age +   temp_apache + ventilated_apache +  d1_diasbp_min + d1_heartrat
y_hat_ct <- predict(fit_ct, edx_test)
mean(edx_test$hospital_death == y_hat_ct)
```

```
## [1] 0.9230324
```

```
sum(y_hat_ct == 1)
```

```
## [1] 116
```

Testing the aids parameter on the gamLoess algorithm.

```
fit_gam <- train(hospital_death ~ age  + temp_apache + ventilated_apache + d1_diasbp_min   + d1_heartra
y_hat_gam <- predict(fit_gam, edx_test)
mean(edx_test$hospital_death == y_hat_gam)
```

```
## [1] 0.9224537
```

```
sum(y_hat_gam == 1)
```

```
## [1] 101
```

Results : In this section we are going to use the validation set as our final test and see what algorithm had the best perfomance and the highest mean.

Testing our final model of the random forest function on the validation set.

```
y_hat_rf <- predict(fit_rf, validation)

mean(validation$hospital_death == y_hat_rf)
```

```
## [1] 0.9218886
```

Testing our final model of the C5.0Tree on the validation set.

```
y_hat_ct <- predict(fit_ct, validation)

mean(validation$hospital_death == y_hat_ct)
```

```
## [1] 0.9165076
```

Testing our final model of the gamloess on the validation set.

```
y_hat_gam <- predict(fit_gam, validation)

mean(validation$hospital_death == y_hat_gam)
```

```
## [1] 0.9213678
```

Conclusions :

The random forest got higher mean but it only limitation is that is very slow to run.

The cs50tree surprisingly doesn't obtain a good perform and a decent mean.

The gamloess algorithm had the second best average of all, but it was very fast to run.

that an algorithm finds a predictor that increases its mean does not mean that in other algorithms it has the same result, for example we have the c50tree and the gamloess.

In a future work we can use more columns of the data table or continue improving the parameters with computers that can support large amounts of information.