Predictive Maintenance

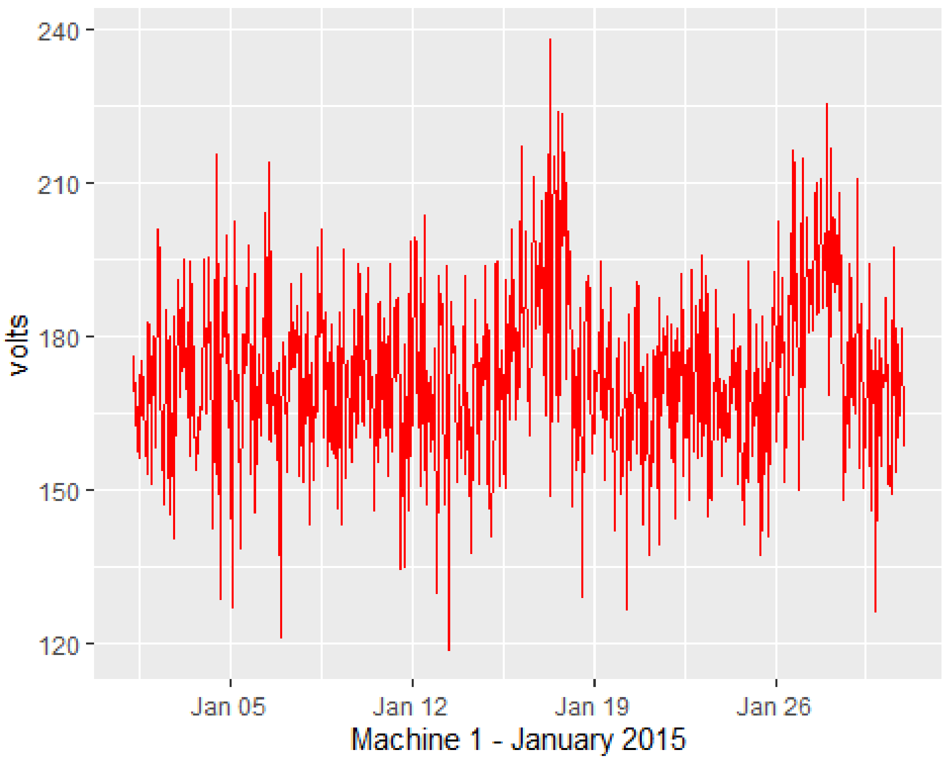
Harsh Mehta

2024-03-21

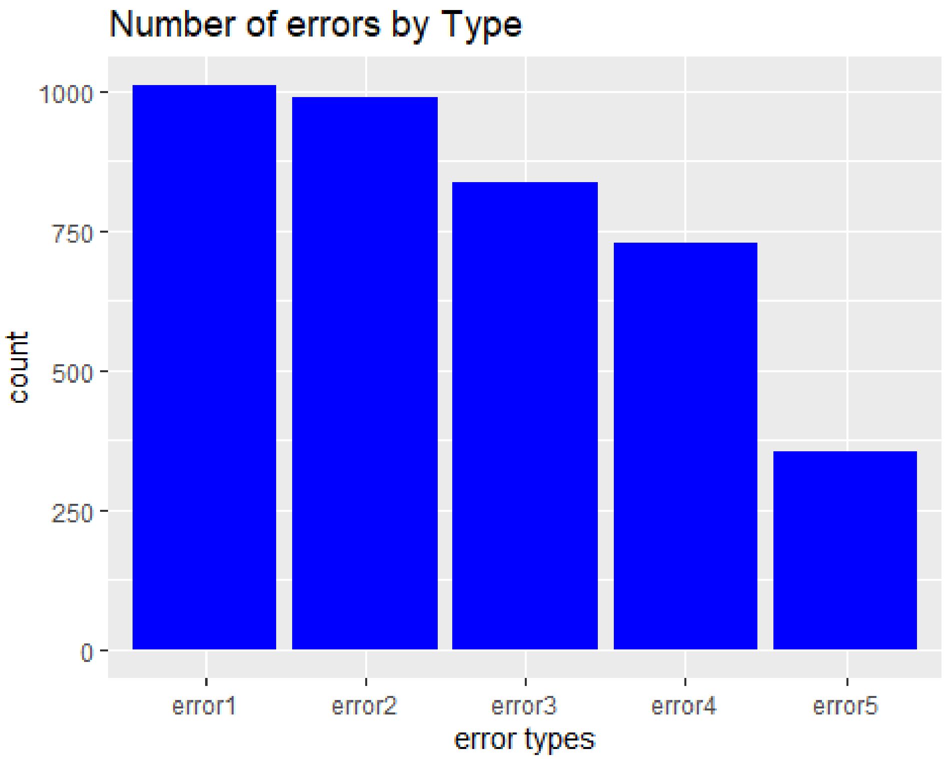
```
library("ggplot2")
library("dplyr")

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library("zoo")

## Warning: package 'zoo' was built under R version 4.2.3

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

library("data.table")

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

library("gbm")

## Warning: package 'gbm' was built under R version 4.2.3

## Loaded gbm 2.1.9

## This version of gbm is no longer under development. Consider transitioning
to gbm3, https://github.com/gbm-developers/gbm3

setwd("C:\\Users\\91797\\Downloads\\OneDrive_2024-01-31\\Case Studies\\Case
Study in R Language")
```

```
telemetry <- read.csv(file='PdM_telemetry.csv')
errors <- read.csv(file='PdM_errors.csv')
maint<-read.csv('PdM_maint.csv')
failures<-read.csv('PdM_failures.csv')
machines<-read.csv('PdM_machines.csv')

#telemetry
#errors
#maint
#machines
#failures

#Step 1 - DATA PRE-PROCESSING
#Telemetry: format datetime field which comes in as.character
telemetry$datetime <- as.POSIXct(telemetry$datetime, format="%Y-%m-%d
%H:%M:%S", tz="UTC")
#Errors: format datetime and errorID fields
errors$datetime <- as.POSIXct(errors$datetime, format="%Y-%m-%d %H:%M:%S",
tz="UTC")
errors$errorID <- as.factor(errors$errorID)
#Maintenance: format datetime and comp fields
maint$datetime <- as.POSIXct(maint$datetime, format="%Y-%m-%d %H:%M:%S",
tz="UTC")
maint$comp <- as.factor(maint$comp)
#Failures: format datetime and failure fields
failures$datetime <- as.POSIXct(failures$datetime, format="%Y-%m-%d
%H:%M:%S", tz="UTC")
failures$failure <- as.factor(failures$failure)
#Machines: format model field
machines$model <- as.factor(machines$model)

str(telemetry)

## 'data.frame':    876100 obs. of  6 variables:
##  $ datetime : POSIXct, format: "2015-01-01 06:00:00" "2015-01-01 07:00:00"
...
##  $ machineID: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ volt     : num  176 163 171 162 158 ...
##  $ rotate   : num  419 403 527 346 435 ...
##  $ pressure : num  113.1 95.5 75.2 109.2 111.9 ...
##  $ vibration: num  45.1 43.4 34.2 41.1 26 ...

str(errors)

## 'data.frame':    3919 obs. of  3 variables:
##  $ datetime : POSIXct, format: "2015-01-03 07:00:00" "2015-01-03 20:00:00"
...
##  $ machineID: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ errorID  : Factor w/ 5 levels "error1","error2",..: 1 3 5 4 4 4 1 2 1 1
...
```

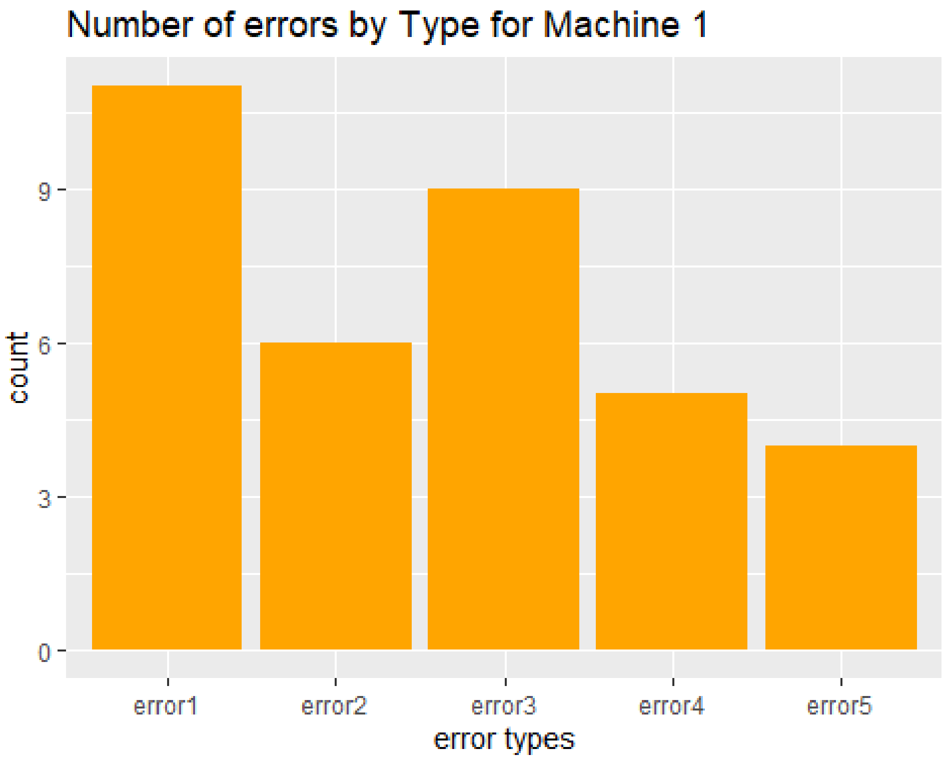
```
str(maint)

## 'data.frame':    3286 obs. of  3 variables:
##  $ datetime : POSIXct, format: "2014-06-01 06:00:00" "2014-07-16 06:00:00"
...
##  $ machineID: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ comp     : Factor w/ 4 levels "comp1","comp2",..: 2 4 3 1 4 1 3 1 4 3
...

str(failures)

## 'data.frame':    761 obs. of  3 variables:
##  $ datetime : POSIXct, format: "2015-01-05 06:00:00" "2015-03-06 06:00:00"
...
##  $ machineID: int  1 1 1 1 1 1 1 2 2 2 ...
##  $ failure  : Factor w/ 4 levels "comp1","comp2",..: 4 1 2 4 4 2 4 1 2 2
...

str(machines)

## 'data.frame':    100 obs. of  3 variables:
##  $ machineID: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ model    : Factor w/ 4 levels "model1","model2",..: 3 4 3 3 3 3 3 3 4 3
...
##  $ age      : int  18 7 8 7 2 7 20 16 7 10 ...

#Telemetry
ggplot(data=telemetry %>%
            filter(machineID==1, datetime>=as.POSIXct("2015-01-01"),
                   datetime<=as.POSIXct("2015-01-31")), aes(x=datetime,
y=volt)) +
            geom_line(color="red")+ labs(x="Machine 1 - January 2015",
y="volts")
```



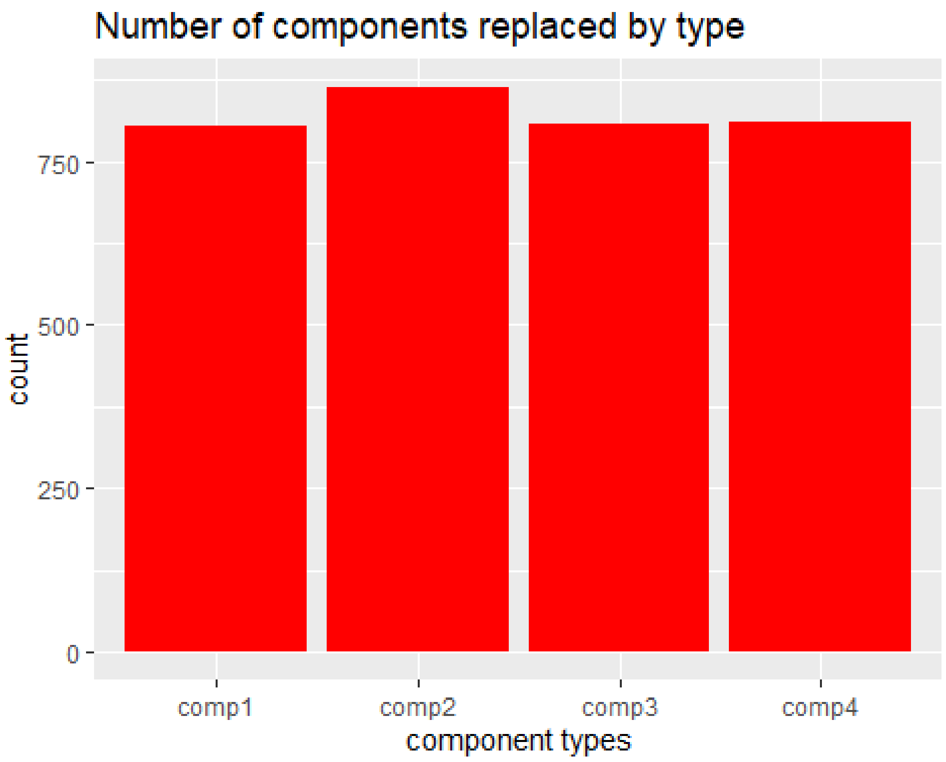
```
#Errors
ggplot(data=errors, aes(x=errorID))+geom_bar(fill="blue",stat="count")+
  labs(title="Number of errors by Type", x="error types")
```



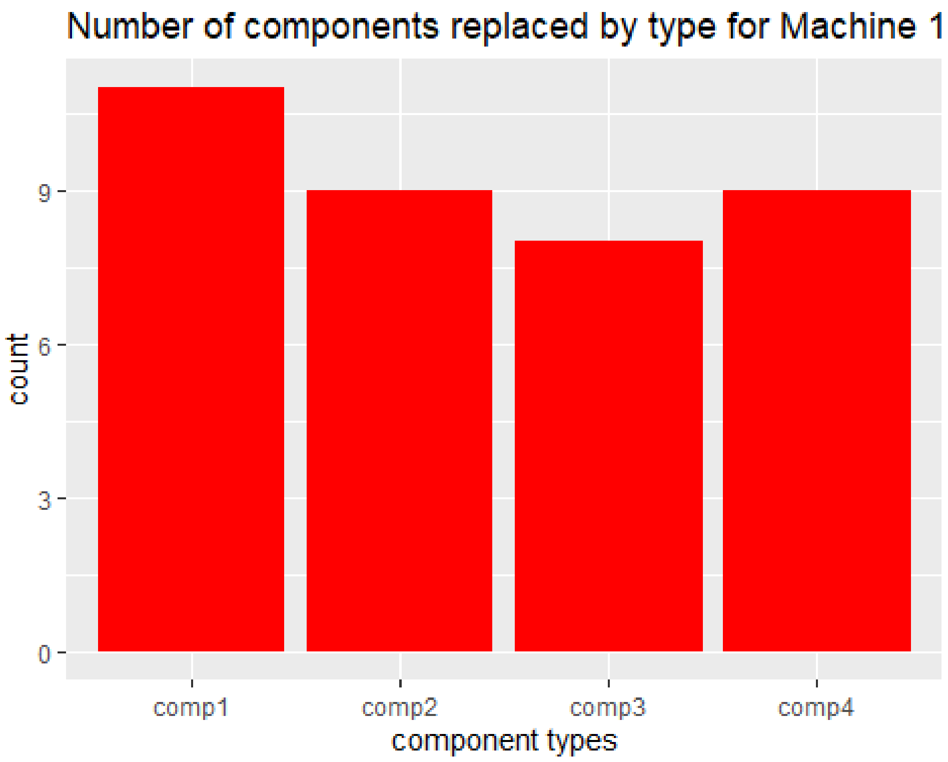
```
ggplot(data=errors %>% filter(machineID==1), aes(x=errorID))+
  geom_bar(fill="orange", stat="count")+
  labs(title="Number of errors by Type for Machine 1", x="error types")
```



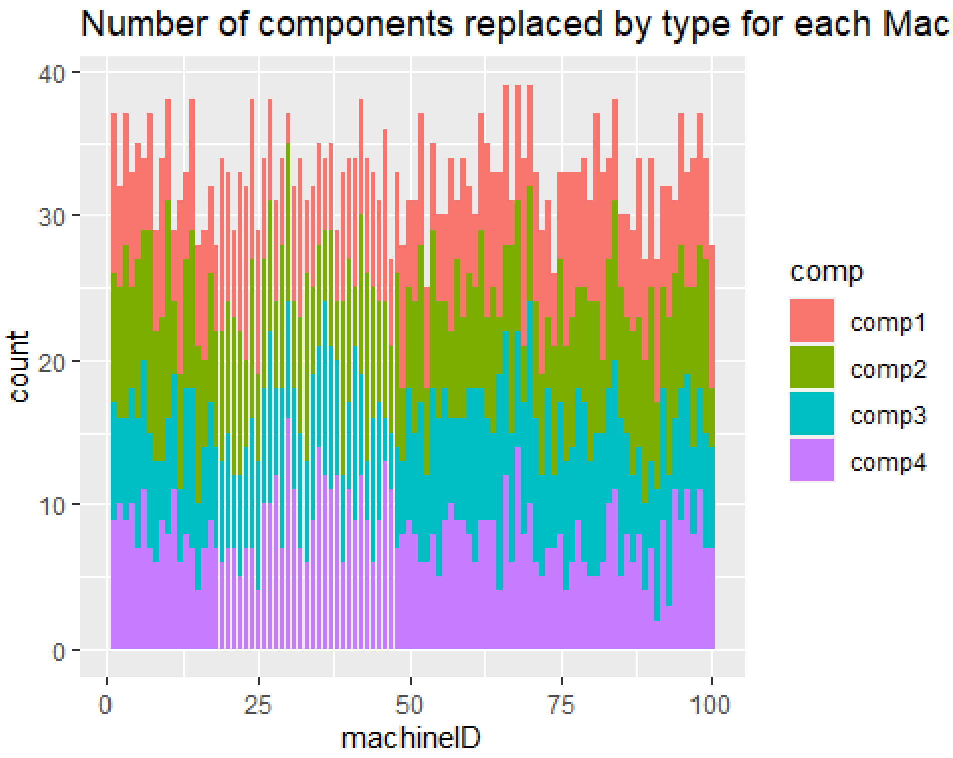
```
#Maintenance
ggplot(data=maint, aes(x=comp))+ geom_bar(fill="red", stat="count")+
  labs(title="Number of components replaced by type", x="component types")
```



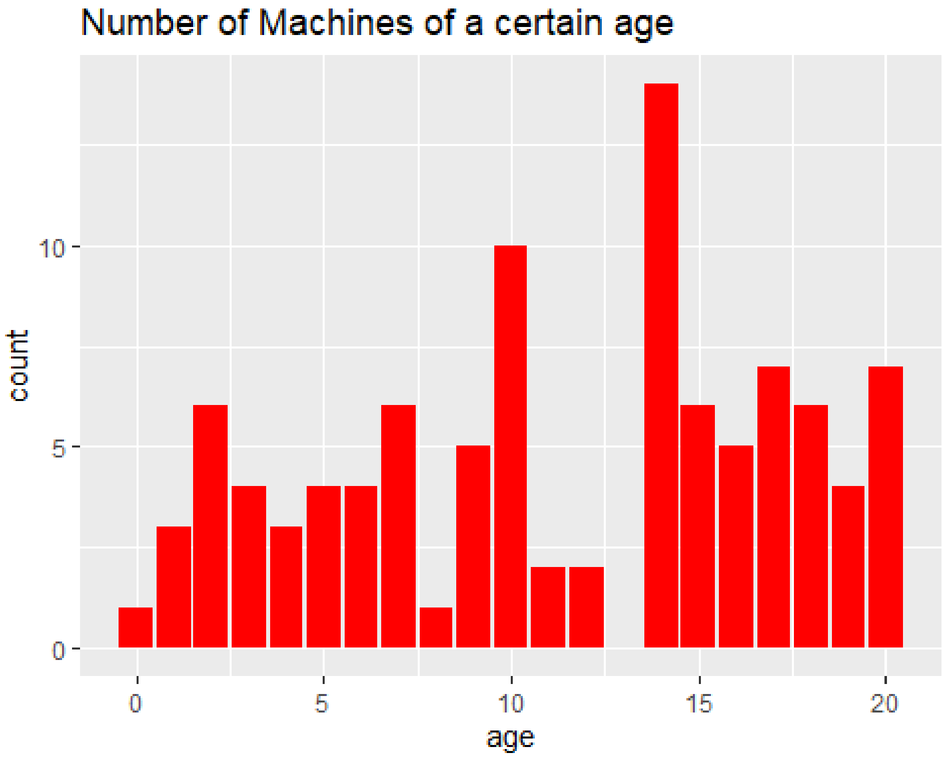
```
ggplot(data=maint %>% filter(machineID==1), aes(x=comp))+
  geom_bar(fill="red", stat="count")+
  labs(title="Number of components replaced by type for Machine 1",
       x="component types")
```



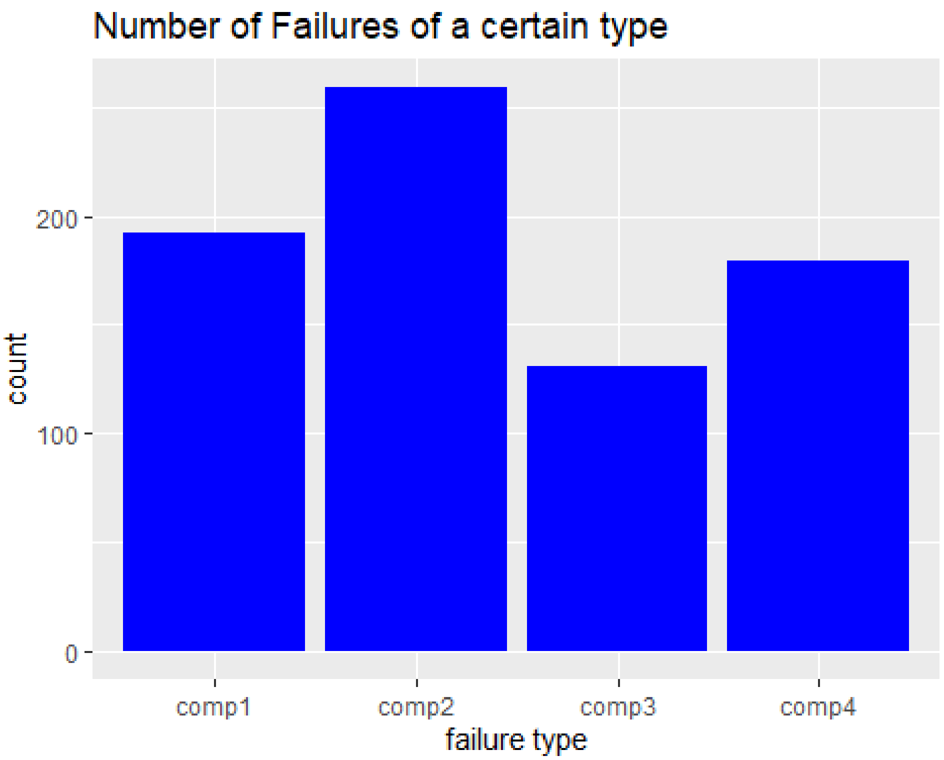
```
ggplot(data=maint, aes(x=machineID))+geom_bar(aes(fill=comp),stat="count")+
  labs(title="Number of components replaced by type for each Machine",
       x="machineID")
```



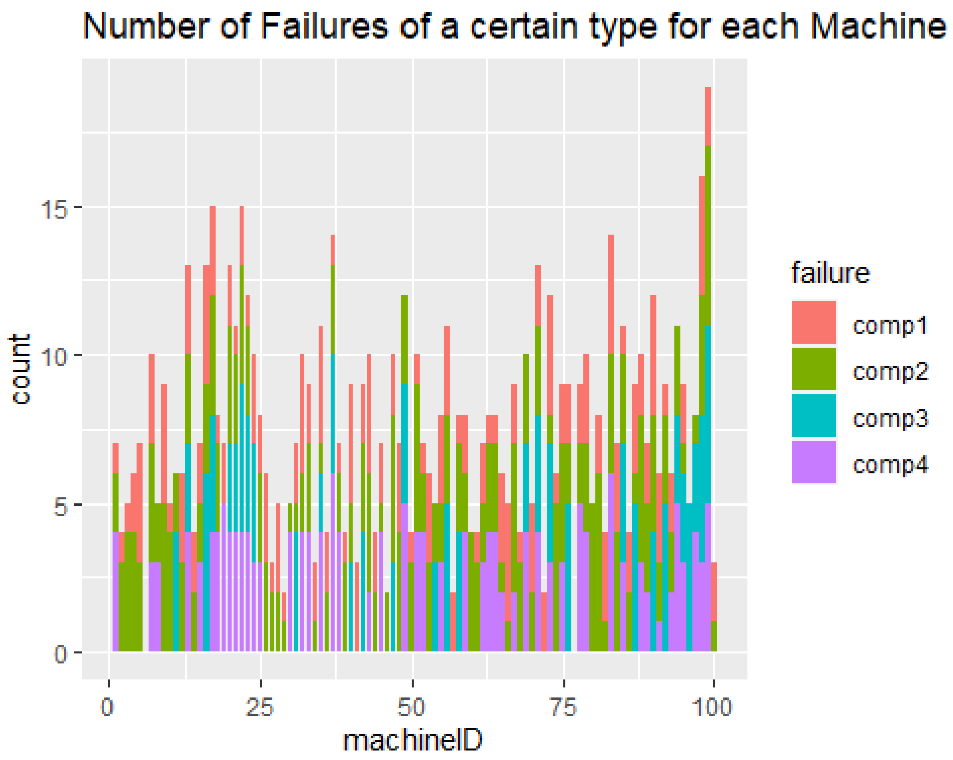
```
#Machines
ggplot(data=machines, aes(x=age))+
  geom_bar(fill="red", stat="count")+
  labs(title="Number of Machines of a certain age", x="age")
```



```
#Failures
ggplot(data=failures, aes(x=failure))+
  geom_bar(fill="blue", stat="count")+
  labs(title="Number of Failures of a certain type", x="failure type")
```



```r
ggplot(data=failures,  aes(x=machineID))+
  geom_bar(aes(fill=failure),stat="count")+
  labs(title="Number of Failures of a certain type for each Machine",
       x="machineID")
```



```
#Step 2 - FEATURE ENGINEERING


#FEATURE ENGINEERING: LAG FEATURES FROM TELEMETRY
telemetrymean<-telemetry %>%
  arrange(machineID,datetime) %>%
  group_by(machineID) %>%
  mutate(voltmean=rollapply(volt, width=3, FUN=mean, align="right", fill=NA,
by=3),
         rotatemean=rollapply(rotate, width=3, FUN=mean, align="right",
fill=NA, by=3),
         pressuremean=rollapply(pressure, width=3, FUN=mean, align="right",
fill=NA, by=3),
         vibrationmean=rollapply(vibration, width=3, FUN=mean, align="right",
fill=NA, by=3)) %>%
  select(datetime, machineID, voltmean, rotatemean, pressuremean,
vibrationmean) %>%
  filter(!is.na(voltmean)) %>%
  ungroup()

head(telemetrymean)

## # A tibble: 6 × 6
##   datetime            machineID voltmean rotatemean pressuremean
vibrationmean
##   <dttm>                  <int>    <dbl>      <dbl>        <dbl>
```

```
<dbl>
## 1 2015-01-01 08:00:00         1     170.       450.         94.6
40.9
## 2 2015-01-01 11:00:00         1     164.       404.        106.
34.3
## 3 2015-01-01 14:00:00         1     168.       436.        108.
41.2
## 4 2015-01-01 17:00:00         1     166.       430.        102.
40.4
## 5 2015-01-01 20:00:00         1     169.       437.         90.9
41.7
## 6 2015-01-01 23:00:00         1     169.       486.         90.4
41.8
```

```
telemetrysd<-telemetry %>%
  arrange(machineID,datetime) %>%
  group_by(machineID) %>%
  mutate(voltsd=rollapply(volt, width=3, FUN=sd, align="right", fill=NA,
by=3),
         rotatesd=rollapply(rotate, width=3, FUN=sd, align="right", fill=NA,
by=3),
         pressuresd=rollapply(pressure, width=3, FUN=sd, align="right",
fill=NA, by=3),
         vibrationsd=rollapply(vibration, width=3, FUN=sd, align="right",
fill=NA, by=3)) %>%
  select(datetime, machineID, voltsd, rotatesd, pressuresd, vibrationsd) %>%
  filter(!is.na(voltsd)) %>%
  ungroup()

head(telemetrysd)
```

```
## # A tibble: 6 × 6
##   datetime            machineID voltsd rotatesd pressuresd vibrationsd
##   <dttm>                  <int>  <dbl>    <dbl>      <dbl>       <dbl>
## 1 2015-01-01 08:00:00         1   6.72     67.8      18.9         5.87
## 2 2015-01-01 11:00:00         1   7.60     50.1       8.56        7.66
## 3 2015-01-01 14:00:00         1  10.1      55.1       5.91        5.17
## 4 2015-01-01 17:00:00         1   4.67     42.0       4.55        2.11
## 5 2015-01-01 20:00:00         1  14.8      47.0       4.24        2.21
## 6 2015-01-01 23:00:00         1  15.9      36.1       4.31        9.39
```

```
telemetrymean_24hours<-telemetry %>%
  arrange(machineID,datetime) %>%
  group_by(machineID) %>%
  mutate(voltmean_24hrs=rollapply(volt, width=24, FUN=mean, align="right",
fill=NA, by=3),
         rotatemean_24hrs=rollapply(rotate, width=24, FUN=mean,
align="right", fill=NA, by=3),
         pressuremean_24hrs=rollapply(pressure, width=24, FUN=mean,
align="right", fill=NA, by=3),
         vibrationmean_24hrs=rollapply(vibration, width=24, FUN=mean,
align="right", fill=NA, by=3)) %>%
  select(datetime, machineID, voltmean_24hrs, rotatemean_24hrs,
pressuremean_24hrs, vibrationmean_24hrs) %>%
  filter(!is.na(voltmean_24hrs)) %>%
  ungroup()

head(telemetrymean_24hours)

## # A tibble: 6 × 6
##   datetime            machineID voltmean_24hrs rotatemean_24hrs
##   <dttm>                  <int>          <dbl>            <dbl>
## 1 2015-01-02 05:00:00         1           170.             445.
## 2 2015-01-02 08:00:00         1           171.             444.
## 3 2015-01-02 11:00:00         1           170.             446.
## 4 2015-01-02 14:00:00         1           170.             447.
## 5 2015-01-02 17:00:00         1           170.             452.
## 6 2015-01-02 20:00:00         1           169.             453.
## # ℹ 2 more variables: pressuremean_24hrs <dbl>, vibrationmean_24hrs <dbl>

telemetrysd_24hours<-telemetry %>%
  arrange(machineID,datetime) %>%
  group_by(machineID) %>%
  mutate(voltsd_24hrs=rollapply(volt, width=24, FUN=sd, align="right",
fill=NA, by=3),
         rotatesd_24hrs=rollapply(rotate, width=24, FUN=sd, align="right",
fill=NA, by=3),
         pressuresd_24hrs=rollapply(pressure, width=24, FUN=sd,
align="right", fill=NA, by=3),
         vibrationsd_24hrs=rollapply(vibration, width=24, FUN=sd,
align="right", fill=NA, by=3)) %>%
  select(datetime, machineID, voltsd_24hrs, rotatesd_24hrs, pressuresd_24hrs,
vibrationsd_24hrs) %>%
  filter(!is.na(voltsd_24hrs)) %>%
  ungroup()

head(telemetrysd_24hours)

## # A tibble: 6 × 6
##   datetime            machineID voltsd_24hrs rotatesd_24hrs
pressuresd_24hrs
##   <dttm>                  <int>        <dbl>          <dbl>
<dbl>
## 1 2015-01-02 05:00:00         1         11.2           48.7
10.1
## 2 2015-01-02 08:00:00         1         12.6           46.9
9.41
## 3 2015-01-02 11:00:00         1         13.3           42.8
9.07
## 4 2015-01-02 14:00:00         1         13.8           42.8
```

```
8.26
## 5 2015-01-02 17:00:00         1          14.8           42.5
8.67
## 6 2015-01-02 20:00:00         1          15.7           41.7
10.6
## # ℹ 1 more variable: vibrationsd_24hrs <dbl>
```

```
telemetryfeat<-data.frame(telemetrymean,telemetrysd[,-c(1:2)])

telemetryfeat_24hours<-
data.frame(telemetrymean_24hours,telemetrysd_24hours[,-c(1:2)])
telemetryfeat_final<-telemetryfeat %>% left_join(telemetryfeat_24hours,
by=c("datetime", "machineID")) %>% filter(!is.na(voltmean_24hrs))

head(telemetryfeat)
```

```
##                datetime machineID voltmean rotatemean pressuremean
vibrationmean
## 1 2015-01-01 08:00:00         1 170.0290   449.5338     94.59212
40.89350
## 2 2015-01-01 11:00:00         1 164.1926   403.9499    105.68742
34.25589
## 3 2015-01-01 14:00:00         1 168.1344   435.7817    107.79371
41.23941
## 4 2015-01-01 17:00:00         1 165.5145   430.4728    101.70329
40.37374
## 5 2015-01-01 20:00:00         1 168.8093   437.1111     90.91106
41.73854
## 6 2015-01-01 23:00:00         1 168.7794   486.2427     90.44647
41.79666
##      voltsd rotatesd pressuresd vibrationsd
## 1  6.721032 67.84960  18.934956    5.874970
## 2  7.596570 50.12045   8.555032    7.662229
## 3 10.124584 55.08473   5.909721    5.169304
## 4  4.673269 42.04728   4.554047    2.106108
## 5 14.752132 47.04861   4.244158    2.207884
## 6 15.901952 36.12955   4.310741    9.390494
```

```
head(telemetryfeat_24hours)
```

```
##                datetime machineID voltmean_24hrs rotatemean_24hrs
## 1 2015-01-02 05:00:00         1       169.7338         445.1799
## 2 2015-01-02 08:00:00         1       170.5257         443.9068
## 3 2015-01-02 11:00:00         1       170.0497         446.4613
## 4 2015-01-02 14:00:00         1       170.3420         447.3553
## 5 2015-01-02 17:00:00         1       170.0606         452.1634
## 6 2015-01-02 20:00:00         1       169.3693         453.3362
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           96.79711            40.38516     11.23312       48.71739
## 2           97.66725            39.78667     12.59195       46.93028
## 3           96.90616            40.01651     13.27734       42.83678
## 4           96.22952            39.92196     13.81716       42.80863
## 5           96.35744            39.99047     14.79287       42.52529
## 6           98.04201            39.53167     15.67479       41.68962
##   pressuresd_24hrs vibrationsd_24hrs
## 1        10.079880          5.853209
## 2         9.406795          6.098173
## 3         9.071472          5.481724
## 4         8.256794          5.862312
## 5         8.669605          5.907157
## 6        10.607947          6.205887
```

```
head(telemetryfeat_final)
```

```
##              datetime machineID voltmean rotatemean pressuremean
vibrationmean
## 1 2015-01-02 05:00:00         1 180.1338   440.6083     94.13797
41.55154
## 2 2015-01-02 08:00:00         1 176.3643   439.3497    101.55321
36.10558
## 3 2015-01-02 11:00:00         1 160.3846   424.3853     99.59872
36.09464
## 4 2015-01-02 14:00:00         1 170.4725   442.9340    102.38059
40.48300
## 5 2015-01-02 17:00:00         1 163.2638   468.9376    102.72665
40.92180
## 6 2015-01-02 20:00:00         1 163.2785   446.4932    104.38758
38.06812
##     voltsd rotatesd pressuresd vibrationsd voltmean_24hrs rotatemean_24hrs
## 1 21.32273 48.77051   2.135684   10.037208       169.7338         445.1799
## 2 18.95221 51.32964  13.789279    6.737739       170.5257         443.9068
## 3 13.04708 13.70250   9.988609    1.639962       170.0497         446.4613
## 4 16.64235 56.29045   3.305739    8.854145       170.3420         447.3553
## 5 17.42469 38.68038   9.105775    3.060781       170.0606         452.1634
## 6 21.58049 41.38096  20.725597    6.932127       169.3693         453.3362
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           96.79711            40.38516     11.23312       48.71739
## 2           97.66725            39.78667     12.59195       46.93028
## 3           96.90616            40.01651     13.27734       42.83678
## 4           96.22952            39.92196     13.81716       42.80863
## 5           96.35744            39.99047     14.79287       42.52529
## 6           98.04201            39.53167     15.67479       41.68962
##   pressuresd_24hrs vibrationsd_24hrs
## 1        10.079880          5.853209
## 2         9.406795          6.098173
## 3         9.071472          5.481724
## 4         8.256794          5.862312
## 5         8.669605          5.907157
## 6        10.607947          6.205887
```

```
head(errors)
```

```
##                datetime machineID errorID
## 1 2015-01-03 07:00:00         1  error1
## 2 2015-01-03 20:00:00         1  error3
## 3 2015-01-04 06:00:00         1  error5
## 4 2015-01-10 15:00:00         1  error4
## 5 2015-01-22 10:00:00         1  error4
## 6 2015-01-25 15:00:00         1  error4
```

```
#FEATURE ENGINEERING: LAG FEATURES FROM ERRORS
#create a column for each error type
errorcount<-errors %>% select(datetime, machineID, errorID) %>%
  mutate(error1=as.integer(errorID=="error1"),
         error2=as.integer(errorID=="error2"),
         error3=as.integer(errorID=="error3"),
         error4=as.integer(errorID=="error4"),
         error5=as.integer(errorID=="error5"))
head(errorcount)
```

```
##                datetime machineID errorID error1 error2 error3 error4 error5
## 1 2015-01-03 07:00:00         1  error1      1      0      0      0      0
## 2 2015-01-03 20:00:00         1  error3      0      0      1      0      0
## 3 2015-01-04 06:00:00         1  error5      0      0      0      0      1
## 4 2015-01-10 15:00:00         1  error4      0      0      0      1      0
## 5 2015-01-22 10:00:00         1  error4      0      0      0      1      0
## 6 2015-01-25 15:00:00         1  error4      0      0      0      1      0
```

```
#sum the duplicate errors in an hour
errorcount_final<-errorcount %>%
  group_by(machineID, datetime) %>%
  summarise(error1sum=sum(error1),
            error2sum=sum(error2),
            error3sum=sum(error3),
            error4sum=sum(error4),
            error5sum=sum(error5)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'machineID'. You can override using
the
## `.groups` argument.
```

```
head(errorcount_final)
```

```
## # A tibble: 6 × 7
##   machineID datetime            error1sum error2sum error3sum error4sum
##       <int> <dttm>                  <int>     <int>     <int>     <int>
## 1         1 2015-01-03 07:00:00         1         0         0         0
## 2         1 2015-01-03 20:00:00         0         0         1         0
## 3         1 2015-01-04 06:00:00         0         0         0         0
## 4         1 2015-01-10 15:00:00         0         0         0         1
## 5         1 2015-01-22 10:00:00         0         0         0         1
## 6         1 2015-01-25 15:00:00         0         0         0         1
## # ℹ 1 more variable: error5sum <int>
```

```
#align errors with telemetry datetime field
errorfeat<-telemetry %>%
  select(datetime, machineID) %>%
  left_join(errorcount_final, by=c("datetime","machineID"))

head(errorfeat)
```

```
##              datetime machineID error1sum error2sum error3sum error4sum
## 1 2015-01-01 06:00:00         1        NA        NA        NA        NA
## 2 2015-01-01 07:00:00         1        NA        NA        NA        NA
## 3 2015-01-01 08:00:00         1        NA        NA        NA        NA
## 4 2015-01-01 09:00:00         1        NA        NA        NA        NA
## 5 2015-01-01 10:00:00         1        NA        NA        NA        NA
## 6 2015-01-01 11:00:00         1        NA        NA        NA        NA
##   error5sum
## 1        NA
## 2        NA
## 3        NA
## 4        NA
## 5        NA
## 6        NA
```

```
#replace missing values
errorfeat[is.na(errorfeat)] <- 0
head(errorfeat)
```

```
##              datetime machineID error1sum error2sum error3sum error4sum
## 1 2015-01-01 06:00:00         1         0         0         0         0
## 2 2015-01-01 07:00:00         1         0         0         0         0
## 3 2015-01-01 08:00:00         1         0         0         0         0
## 4 2015-01-01 09:00:00         1         0         0         0         0
## 5 2015-01-01 10:00:00         1         0         0         0         0
## 6 2015-01-01 11:00:00         1         0         0         0         0
##   error5sum
## 1         0
## 2         0
## 3         0
## 4         0
## 5         0
## 6         0
```

```
#count the number of errors of different types in the last 24 hours, for
every 3 hours
errorfeat_final<-errorfeat %>%
  arrange(machineID, datetime) %>%
  group_by(machineID) %>%
  mutate(error1count=rollapply(error1sum, width=24, FUN=sum, align="right",
fill=NA, by=3),
         error2count=rollapply(error2sum, width=24, FUN=sum, align="right",
fill=NA, by=3),
         error3count=rollapply(error3sum, width=24, FUN=sum, align="right",
fill=NA, by=3),
         error4count=rollapply(error4sum, width=24, FUN=sum, align="right",
fill=NA, by=3),
         error5count=rollapply(error5sum, width=24, FUN=sum, align="right",
fill=NA, by=3)) %>%
  select(datetime, machineID, error1count, error2count, error3count,
error4count, error5count) %>%
  filter(!is.na(error1count)) %>%
  ungroup()

head(errorfeat_final)
```

```
## # A tibble: 6 × 7
##   datetime            machineID error1count error2count error3count
error4count
##   <dttm>                  <int>       <dbl>       <dbl>       <dbl>
<dbl>
## 1 2015-01-02 05:00:00         1           0           0           0
0
## 2 2015-01-02 08:00:00         1           0           0           0
0
## 3 2015-01-02 11:00:00         1           0           0           0
0
## 4 2015-01-02 14:00:00         1           0           0           0
0
## 5 2015-01-02 17:00:00         1           0           0           0
0
## 6 2015-01-02 20:00:00         1           0           0           0
0
## # ℹ 1 more variable: error5count <dbl>
```

```
head(failures)
```

```
##              datetime machineID failure
## 1 2015-01-05 06:00:00         1   comp4
## 2 2015-03-06 06:00:00         1   comp1
## 3 2015-04-20 06:00:00         1   comp2
## 4 2015-06-19 06:00:00         1   comp4
## 5 2015-09-02 06:00:00         1   comp4
## 6 2015-10-17 06:00:00         1   comp2
```

```
head(maint)
```

```
##              datetime machineID  comp
## 1 2014-06-01 06:00:00         1 comp2
## 2 2014-07-16 06:00:00         1 comp4
## 3 2014-07-31 06:00:00         1 comp3
## 4 2014-12-13 06:00:00         1 comp1
## 5 2015-01-05 06:00:00         1 comp4
## 6 2015-01-05 06:00:00         1 comp1
```

```
#FEATURE ENGINEERING: NUMBER OF DAYS SINCE LAST REPLACEMENT FROM MAINTENANCE
#create a binary column for each component. 1 if a replacement occured, 0 if not.
comprep <- maint %>%
    select(datetime, machineID, comp) %>%
    mutate(comp1=as.integer(comp=="comp1"),
           comp2=as.integer(comp=="comp2"),
           comp3=as.integer(comp=="comp3"),
           comp4=as.integer(comp=="comp4")) %>%
  select(-comp)
head(comprep)
```

```
##              datetime machineID comp1 comp2 comp3 comp4
## 1 2014-06-01 06:00:00         1     0     1     0     0
## 2 2014-07-16 06:00:00         1     0     0     0     1
## 3 2014-07-31 06:00:00         1     0     0     1     0
## 4 2014-12-13 06:00:00         1     1     0     0     0
## 5 2015-01-05 06:00:00         1     0     0     0     1
## 6 2015-01-05 06:00:00         1     1     0     0     0
```

```
comprep<-as.data.table(comprep)
setkey(comprep,machineID, datetime)

#separate different component type replacements into different tables
comp1rep<-comprep[comp1==1, .(machineID, datetime, lastrepcomp1=datetime)]
comp2rep<-comprep[comp2==1, .(machineID, datetime, lastrepcomp2=datetime)]
comp3rep<-comprep[comp3==1, .(machineID, datetime, lastrepcomp3=datetime)]
comp4rep<-comprep[comp4==1, .(machineID, datetime, lastrepcomp4=datetime)]

#use telemetry feature table datetime and machineID to be matched with replacements
compdate <- as.data.table(telemetryfeat_final[,c(1:2)])
setkey(compdate,machineID, datetime)

#data.table rolling match will attach the latest record from the component replacement tables
#to the telemetry date time and machineID
comp1feat<-comp1rep[compdate[,.(machineID, datetime)], roll=TRUE]
comp1feat$sincelastcomp1<-as.numeric(difftime(comp1feat$datetime, comp1feat$lastrepcomp1, units="days"))

comp2feat<-comp2rep[compdate[,.(machineID, datetime)], roll=TRUE]
comp2feat$sincelastcomp2<-as.numeric(difftime(comp2feat$datetime, comp2feat$lastrepcomp2, units="days"))

comp3feat<-comp3rep[compdate[,.(machineID, datetime)], roll=TRUE]
comp3feat$sincelastcomp3<-as.numeric(difftime(comp3feat$datetime, comp3feat$lastrepcomp3, units="days"))
```

```
comp4feat<-comp4rep[compdate[,.(machineID, datetime)], roll=TRUE]
comp4feat$sincelastcomp4<-as.numeric(difftime(comp4feat$datetime,
comp4feat$lastrepcomp4, units="days"))

#merge all tables
compfeat_final<-data.frame(compdate,
comp1feat[,.(sincelastcomp1)],comp2feat[,.(sincelastcomp2)],comp3feat[,.(sinc
elastcomp3)],comp4feat[,.(sincelastcomp4)])

head(compfeat_final)
```

```
##              datetime machineID sincelastcomp1 sincelastcomp2
sincelastcomp3
## 1 2015-01-02 05:00:00         1       19.95833       214.9583
154.9583
## 2 2015-01-02 08:00:00         1       20.08333       215.0833
155.0833
## 3 2015-01-02 11:00:00         1       20.20833       215.2083
155.2083
## 4 2015-01-02 14:00:00         1       20.33333       215.3333
155.3333
## 5 2015-01-02 17:00:00         1       20.45833       215.4583
155.4583
## 6 2015-01-02 20:00:00         1       20.58333       215.5833
155.5833
##   sincelastcomp4
## 1       169.9583
## 2       170.0833
## 3       170.2083
## 4       170.3333
## 5       170.4583
## 6       170.5833
```

```
head(machines)
```

```
##   machineID  model age
## 1         1 model3  18
## 2         2 model4   7
## 3         3 model3   8
## 4         4 model3   7
## 5         5 model3   2
## 6         6 model3   7
```

```
#FEATURE ENGINEERING: MERGE TELEMETRYFEAT_FINAL, ERRORFEAT_FINAL
  finalfeat <- data.frame(telemetryfeat_final, errorfeat_final[,-c(1:2)])

#MERGE finalfeat con COMPFEAT_FINAL and machines features
  finalfeat <- finalfeat %>%
    left_join(compfeat_final, by=c("datetime", "machineID")) %>%
    left_join(machines, by=c("machineID"))
```

```
str(finalfeat)
```

```
## 'data.frame':    291300 obs. of  29 variables:
##  $ datetime           : POSIXct, format: "2015-01-02 05:00:00" "2015-01-02
08:00:00" ...
##  $ machineID          : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ voltmean           : num  180 176 160 170 163 ...
##  $ rotatemean         : num  441 439 424 443 469 ...
##  $ pressuremean       : num  94.1 101.6 99.6 102.4 102.7 ...
##  $ vibrationmean      : num  41.6 36.1 36.1 40.5 40.9 ...
##  $ voltsd             : num  21.3 19 13 16.6 17.4 ...
##  $ rotatesd           : num  48.8 51.3 13.7 56.3 38.7 ...
##  $ pressuresd         : num  2.14 13.79 9.99 3.31 9.11 ...
##  $ vibrationsd        : num  10.04 6.74 1.64 8.85 3.06 ...
##  $ voltmean_24hrs     : num  170 171 170 170 170 ...
##  $ rotatemean_24hrs   : num  445 444 446 447 452 ...
##  $ pressuremean_24hrs : num  96.8 97.7 96.9 96.2 96.4 ...
##  $ vibrationmean_24hrs: num  40.4 39.8 40 39.9 40 ...
##  $ voltsd_24hrs       : num  11.2 12.6 13.3 13.8 14.8 ...
##  $ rotatesd_24hrs     : num  48.7 46.9 42.8 42.8 42.5 ...
##  $ pressuresd_24hrs   : num  10.08 9.41 9.07 8.26 8.67 ...
##  $ vibrationsd_24hrs  : num  5.85 6.1 5.48 5.86 5.91 ...
##  $ error1count        : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ error2count        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ error3count        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ error4count        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ error5count        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ sincelastcomp1     : num  20 20.1 20.2 20.3 20.5 ...
##  $ sincelastcomp2     : num  215 215 215 215 215 ...
##  $ sincelastcomp3     : num  155 155 155 155 155 ...
##  $ sincelastcomp4     : num  170 170 170 170 170 ...
##  $ model              : Factor w/ 4 levels "model1","model2",..: 3 3 3 3 3
3 3 3 3 3 ...
##  $ age                : int  18 18 18 18 18 18 18 18 18 18 ...
```

```
head(failures)
```

```
##              datetime machineID failure
## 1 2015-01-05 06:00:00         1   comp4
## 2 2015-03-06 06:00:00         1   comp1
## 3 2015-04-20 06:00:00         1   comp2
## 4 2015-06-19 06:00:00         1   comp4
## 5 2015-09-02 06:00:00         1   comp4
## 6 2015-10-17 06:00:00         1   comp2
```

```
head(finalfeat)
```

```
##              datetime machineID voltmean rotatemean pressuremean
vibrationmean
```

```
## 1 2015-01-02 05:00:00         1 180.1338   440.6083      94.13797
41.55154
## 2 2015-01-02 08:00:00         1 176.3643   439.3497     101.55321
36.10558
## 3 2015-01-02 11:00:00         1 160.3846   424.3853      99.59872
36.09464
## 4 2015-01-02 14:00:00         1 170.4725   442.9340     102.38059
40.48300
## 5 2015-01-02 17:00:00         1 163.2638   468.9376     102.72665
40.92180
## 6 2015-01-02 20:00:00         1 163.2785   446.4932     104.38758
38.06812
##     voltsd rotatesd pressuresd vibrationsd voltmean_24hrs rotatemean_24hrs
## 1 21.32273 48.77051   2.135684   10.037208       169.7338         445.1799
## 2 18.95221 51.32964  13.789279    6.737739       170.5257         443.9068
## 3 13.04708 13.70250   9.988609    1.639962       170.0497         446.4613
## 4 16.64235 56.29045   3.305739    8.854145       170.3420         447.3553
## 5 17.42469 38.68038   9.105775    3.060781       170.0606         452.1634
## 6 21.58049 41.38096  20.725597    6.932127       169.3693         453.3362
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           96.79711            40.38516     11.23312       48.71739
## 2           97.66725            39.78667     12.59195       46.93028
## 3           96.90616            40.01651     13.27734       42.83678
## 4           96.22952            39.92196     13.81716       42.80863
## 5           96.35744            39.99047     14.79287       42.52529
## 6           98.04201            39.53167     15.67479       41.68962
##   pressuresd_24hrs vibrationsd_24hrs error1count error2count error3count
## 1        10.079880          5.853209           0           0           0
## 2         9.406795          6.098173           0           0           0
## 3         9.071472          5.481724           0           0           0
## 4         8.256794          5.862312           0           0           0
## 5         8.669605          5.907157           0           0           0
## 6        10.607947          6.205887           0           0           0
##   error4count error5count sincelastcomp1 sincelastcomp2 sincelastcomp3
## 1           0           0       19.95833       214.9583       154.9583
## 2           0           0       20.08333       215.0833       155.0833
## 3           0           0       20.20833       215.2083       155.2083
## 4           0           0       20.33333       215.3333       155.3333
## 5           0           0       20.45833       215.4583       155.4583
## 6           0           0       20.58333       215.5833       155.5833
##   sincelastcomp4  model age
## 1       169.9583 model3  18
## 2       170.0833 model3  18
## 3       170.2083 model3  18
## 4       170.3333 model3  18
## 5       170.4583 model3  18
## 6       170.5833 model3  18

#Step 3 - LABELING
# The prediction problem for this example scenario is to estimate the
```

```
probability
# that a machine will fail in the near future due to a failure
# of a certain component. More specifically, the goal is to compute the
probability
# that a machine will fail in the next 24 hours due to a certain
# component failure (component 1, 2, 3, or 4).
# Below, a categorical failure feature is created to serve as the label.
# All records within a 24 hour window before a failure of component 1 have
failure=comp1,
# and so on for components 2, 3, and 4;
# all records not within 24 hours of a component failure have failure=none.

# left join final features with failures on machineID then mutate a column
for datetime difference
# filter date difference for the prediction horizon which is 24 hours

labeled <- left_join(finalfeat, failures, by = c("machineID")) %>%
    mutate(datediff = difftime(datetime.y, datetime.x, units = "hours")) %>%
    filter(datediff <= 24, datediff >= 0)
head(labeled)
```

```
##             datetime.x machineID voltmean rotatemean pressuremean
vibrationmean
## 1 2015-01-04 08:00:00         1 166.2818   453.7878    106.18758
51.99008
## 2 2015-01-04 11:00:00         1 175.4121   445.4506    100.88736
54.25153
## 3 2015-01-04 14:00:00         1 157.3477   451.8821    101.28938
48.60269
## 4 2015-01-04 17:00:00         1 176.4506   446.0331     84.52155
47.63884
## 5 2015-01-04 20:00:00         1 190.3258   422.6926    107.39323
49.55286
## 6 2015-01-04 23:00:00         1 169.9851   458.9294     91.49436
54.88202
##      voltsd  rotatesd pressuresd vibrationsd voltmean_24hrs
rotatemean_24hrs
## 1 24.276228 23.621315  11.176731    3.394073       171.8041
444.4782
## 2 34.918687 11.001625  10.580336    2.921501       171.9427
445.6367
## 3 24.617739 28.950883   9.966729    2.356486       169.5803
445.6662
## 4  8.071400 76.511343   2.636879    4.108621       171.8325
444.6828
## 5  8.390777  7.176553   4.262645    7.598552       175.3247
440.7518
## 6  9.451483 12.052752   3.685906    6.621183       174.7924
448.7432
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           101.1983            52.60454     14.67779       37.12815
## 2           102.0476            53.01104     17.05520       36.30528
## 3           102.4460            51.68659     18.69166       36.11981
## 4           100.6514            51.78495     18.36861       42.00840
## 5           102.4942            51.19636     19.02541       38.91638
## 6           101.4523            52.19027     19.22466       34.00803
##   pressuresd_24hrs vibrationsd_24hrs error1count error2count error3count
## 1        10.440118          5.794546           0           0           1
## 2        10.310175          5.752299           0           0           1
## 3         9.579414          5.053566           0           0           1
## 4        10.860645          5.045402           0           0           1
## 5        10.564317          5.261867           0           0           0
## 6        10.807630          5.081258           0           0           0
##   error4count error5count sincelastcomp1 sincelastcomp2 sincelastcomp3
## 1           0           1       22.08333       217.0833       157.0833
## 2           0           1       22.20833       217.2083       157.2083
## 3           0           1       22.33333       217.3333       157.3333
## 4           0           1       22.45833       217.4583       157.4583
## 5           0           1       22.58333       217.5833       157.5833
## 6           0           1       22.70833       217.7083       157.7083
##   sincelastcomp4  model age          datetime.y failure datediff
## 1       172.0833 model3  18 2015-01-05 06:00:00   comp4 22 hours
## 2       172.2083 model3  18 2015-01-05 06:00:00   comp4 19 hours
## 3       172.3333 model3  18 2015-01-05 06:00:00   comp4 16 hours
## 4       172.4583 model3  18 2015-01-05 06:00:00   comp4 13 hours
## 5       172.5833 model3  18 2015-01-05 06:00:00   comp4 10 hours
## 6       172.7083 model3  18 2015-01-05 06:00:00   comp4  7 hours
```

```
# left join labels to final features and fill NA's with "none" indicating no
failure
labeledfeatures <- left_join(finalfeat, labeled %>% select(datetime.x,
machineID, failure),
                               by = c("datetime" = "datetime.x",
"machineID")) %>%
                               arrange(machineID,datetime)

levels(labeledfeatures$failure) <- c(levels(labeledfeatures$failure), "none")
labeledfeatures$failure[is.na(labeledfeatures$failure)]<-"none"
head(labeledfeatures)
```

```
##              datetime machineID voltmean rotatemean pressuremean
vibrationmean
## 1 2015-01-02 05:00:00         1 180.1338   440.6083     94.13797
41.55154
## 2 2015-01-02 08:00:00         1 176.3643   439.3497    101.55321
36.10558
## 3 2015-01-02 11:00:00         1 160.3846   424.3853     99.59872
36.09464
## 4 2015-01-02 14:00:00         1 170.4725   442.9340    102.38059
40.48300
```

```
## 5 2015-01-02 17:00:00         1 163.2638   468.9376    102.72665
40.92180
## 6 2015-01-02 20:00:00         1 163.2785   446.4932    104.38758
38.06812
##     voltsd rotatesd pressuresd vibrationsd voltmean_24hrs rotatemean_24hrs
## 1 21.32273 48.77051   2.135684   10.037208       169.7338         445.1799
## 2 18.95221 51.32964  13.789279    6.737739       170.5257         443.9068
## 3 13.04708 13.70250   9.988609    1.639962       170.0497         446.4613
## 4 16.64235 56.29045   3.305739    8.854145       170.3420         447.3553
## 5 17.42469 38.68038   9.105775    3.060781       170.0606         452.1634
## 6 21.58049 41.38096  20.725597    6.932127       169.3693         453.3362
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           96.79711            40.38516     11.23312       48.71739
## 2           97.66725            39.78667     12.59195       46.93028
## 3           96.90616            40.01651     13.27734       42.83678
## 4           96.22952            39.92196     13.81716       42.80863
## 5           96.35744            39.99047     14.79287       42.52529
## 6           98.04201            39.53167     15.67479       41.68962
##   pressuresd_24hrs vibrationsd_24hrs error1count error2count error3count
## 1        10.079880          5.853209           0           0           0
## 2         9.406795          6.098173           0           0           0
## 3         9.071472          5.481724           0           0           0
## 4         8.256794          5.862312           0           0           0
## 5         8.669605          5.907157           0           0           0
## 6        10.607947          6.205887           0           0           0
##   error4count error5count sincelastcomp1 sincelastcomp2 sincelastcomp3
## 1           0           0       19.95833       214.9583       154.9583
## 2           0           0       20.08333       215.0833       155.0833
## 3           0           0       20.20833       215.2083       155.2083
## 4           0           0       20.33333       215.3333       155.3333
## 5           0           0       20.45833       215.4583       155.4583
## 6           0           0       20.58333       215.5833       155.5833
##   sincelastcomp4  model age failure
## 1       169.9583 model3  18    none
## 2       170.0833 model3  18    none
## 3       170.2083 model3  18    none
## 4       170.3333 model3  18    none
## 5       170.4583 model3  18    none
## 6       170.5833 model3  18    none
```

```
head(labeledfeatures)
```

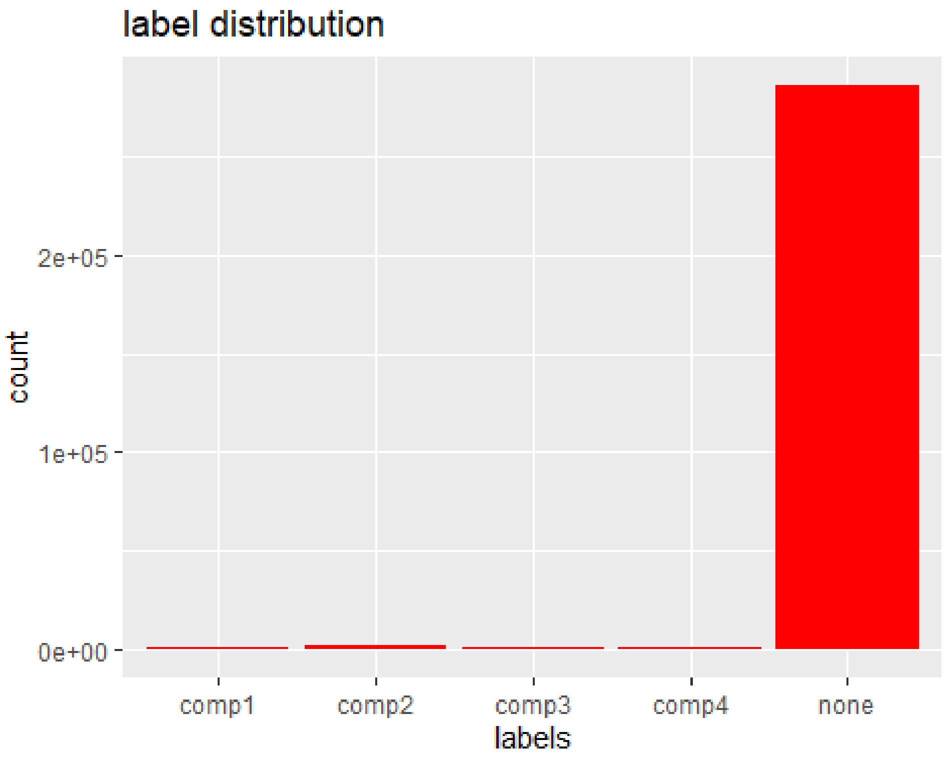
```
##                datetime machineID voltmean rotatemean pressuremean
vibrationmean
## 1 2015-01-02 05:00:00         1 180.1338   440.6083     94.13797
41.55154
## 2 2015-01-02 08:00:00         1 176.3643   439.3497    101.55321
36.10558
## 3 2015-01-02 11:00:00         1 160.3846   424.3853     99.59872
36.09464
```

```
## 4 2015-01-02 14:00:00         1 170.4725   442.9340    102.38059
40.48300
## 5 2015-01-02 17:00:00         1 163.2638   468.9376    102.72665
40.92180
## 6 2015-01-02 20:00:00         1 163.2785   446.4932    104.38758
38.06812
##     voltsd rotatesd pressuresd vibrationsd voltmean_24hrs rotatemean_24hrs
## 1 21.32273 48.77051   2.135684   10.037208       169.7338         445.1799
## 2 18.95221 51.32964  13.789279    6.737739       170.5257         443.9068
## 3 13.04708 13.70250   9.988609    1.639962       170.0497         446.4613
## 4 16.64235 56.29045   3.305739    8.854145       170.3420         447.3553
## 5 17.42469 38.68038   9.105775    3.060781       170.0606         452.1634
## 6 21.58049 41.38096  20.725597    6.932127       169.3693         453.3362
##   pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs rotatesd_24hrs
## 1           96.79711            40.38516     11.23312       48.71739
## 2           97.66725            39.78667     12.59195       46.93028
## 3           96.90616            40.01651     13.27734       42.83678
## 4           96.22952            39.92196     13.81716       42.80863
## 5           96.35744            39.99047     14.79287       42.52529
## 6           98.04201            39.53167     15.67479       41.68962
##   pressuresd_24hrs vibrationsd_24hrs error1count error2count error3count
## 1        10.079880          5.853209           0           0           0
## 2         9.406795          6.098173           0           0           0
## 3         9.071472          5.481724           0           0           0
## 4         8.256794          5.862312           0           0           0
## 5         8.669605          5.907157           0           0           0
## 6        10.607947          6.205887           0           0           0
##   error4count error5count sincelastcomp1 sincelastcomp2 sincelastcomp3
## 1           0           0       19.95833       214.9583       154.9583
## 2           0           0       20.08333       215.0833       155.0833
## 3           0           0       20.20833       215.2083       155.2083
## 4           0           0       20.33333       215.3333       155.3333
## 5           0           0       20.45833       215.4583       155.4583
## 6           0           0       20.58333       215.5833       155.5833
##   sincelastcomp4  model age failure
## 1       169.9583 model3  18    none
## 2       170.0833 model3  18    none
## 3       170.2083 model3  18    none
## 4       170.3333 model3  18    none
## 5       170.4583 model3  18    none
## 6       170.5833 model3  18    none
```

```
#number of records with failure different from none
length(which(labeledfeatures$failure!="none" ))
```

```
## [1] 5923
```

```
# label distribution after features are labeled - the class imbalance problem
ggplot(labeledfeatures, aes(x=failure)) +
  geom_bar(fill="red") +
  labs(title = "label distribution", x = "labels")
```



```
#Step 4 - Modelling
#split at 2015-08-01 01:00:00, first 8 months train, last 4 month test
trainingdata1 <- labeledfeatures[labeledfeatures$datetime < "2015-07-31
01:00:00",]
testingdata1 <-labeledfeatures[labeledfeatures$datetime > "2015-08-01
01:00:00",]

#split at 2015-09-01 01:00:00, first 9 months train, last 3 month test
trainingdata2 <- labeledfeatures[labeledfeatures$datetime < "2015-08-31
01:00:00",]
testingdata2 <-labeledfeatures[labeledfeatures$datetime > "2015-09-01
01:00:00",]

#split at 2015-10-01 01:00:00, first 8 months train, last 4 month test
trainingdata3 <- labeledfeatures[labeledfeatures$datetime < "2015-09-30
01:00:00",]
testingdata3 <-labeledfeatures[labeledfeatures$datetime > "2015-10-01
01:00:00",]



#create the training formula
trainformula <-as.formula(paste('failure',
paste(names(labeledfeatures)[c(3:29)], collapse=' + '), sep=' ~ '))
trainformula
```

```
## failure ~ voltmean + rotatemean + pressuremean + vibrationmean +
##     voltsd + rotatesd + pressuresd + vibrationsd + voltmean_24hrs +
##     rotatemean_24hrs + pressuremean_24hrs + vibrationmean_24hrs +
##     voltsd_24hrs + rotatesd_24hrs + pressuresd_24hrs + vibrationsd_24hrs +
##     error1count + error2count + error3count + error4count + error5count +
##     sincelastcomp1 + sincelastcomp2 + sincelastcomp3 + sincelastcomp4 +
##     model + age
```

```
set.seed(1234)

gbm_model1 <- gbm(formula=trainformula, data= trainingdata1,
distribution="multinomial", n.trees =50, interaction.depth =5, shrinkage
=0.1)
```

```
## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your
own
## risk.
```

```
gbm_model2 <- gbm(formula=trainformula, data= trainingdata2,
distribution="multinomial", n.trees =50, interaction.depth =5, shrinkage
=0.1)
```

```
## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your
own
## risk.
```

```
gbm_model3 <- gbm(formula=trainformula, data= trainingdata3,
distribution="multinomial", n.trees =50, interaction.depth =5, shrinkage
=0.1)
```

```
## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your
own
## risk.
```

```
#print the relative influence of variables for the three models
gbm_model1
```

```
## gbm(formula = trainformula, distribution = "multinomial", data =
trainingdata1,
##     n.trees = 50, interaction.depth = 5, shrinkage = 0.1)
## A gradient boosted model with multinomial loss function.
## 50 iterations were performed.
## There were 27 predictors of which 27 had non-zero influence.
```

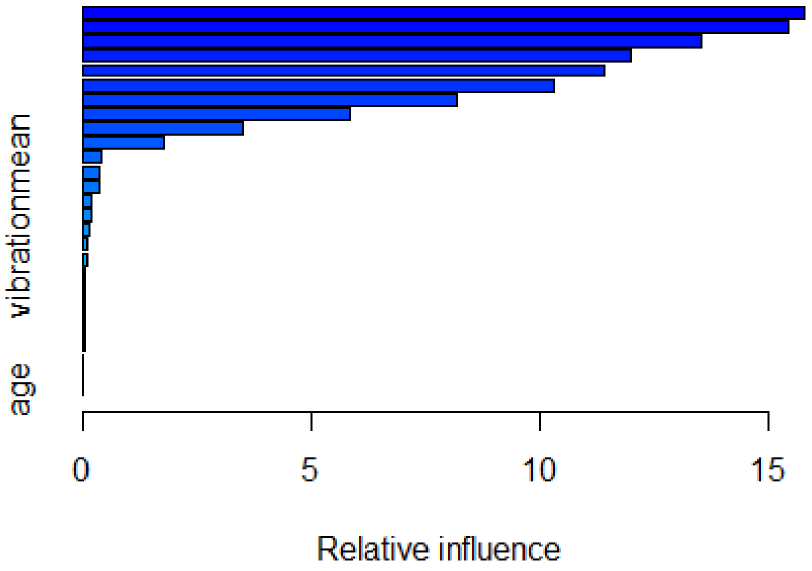
```
gbm_model2
```

```
## gbm(formula = trainformula, distribution = "multinomial", data =
trainingdata2,
##     n.trees = 50, interaction.depth = 5, shrinkage = 0.1)
## A gradient boosted model with multinomial loss function.
## 50 iterations were performed.
## There were 27 predictors of which 27 had non-zero influence.
```

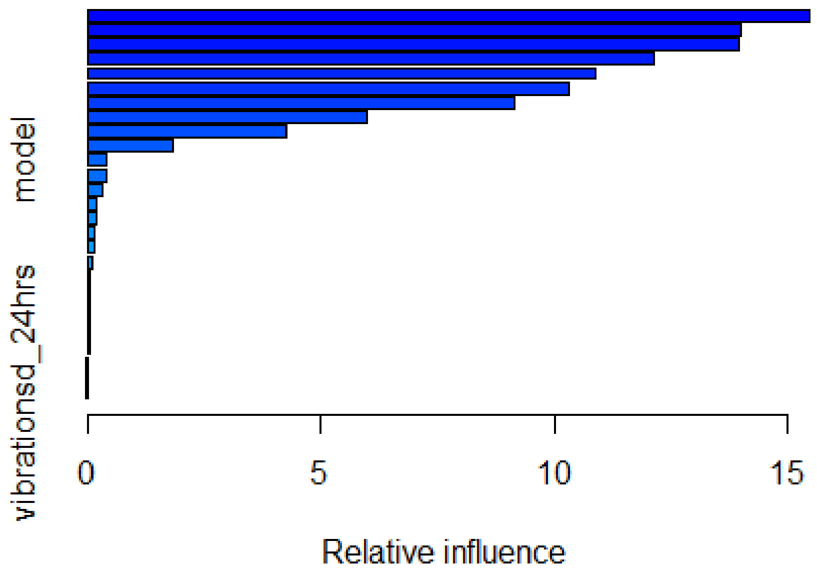
```
gbm_model3
```

```
## gbm(formula = trainformula, distribution = "multinomial", data =
trainingdata3,
##     n.trees = 50, interaction.depth = 5, shrinkage = 0.1)
## A gradient boosted model with multinomial loss function.
## 50 iterations were performed.
## There were 27 predictors of which 27 had non-zero influence.
```

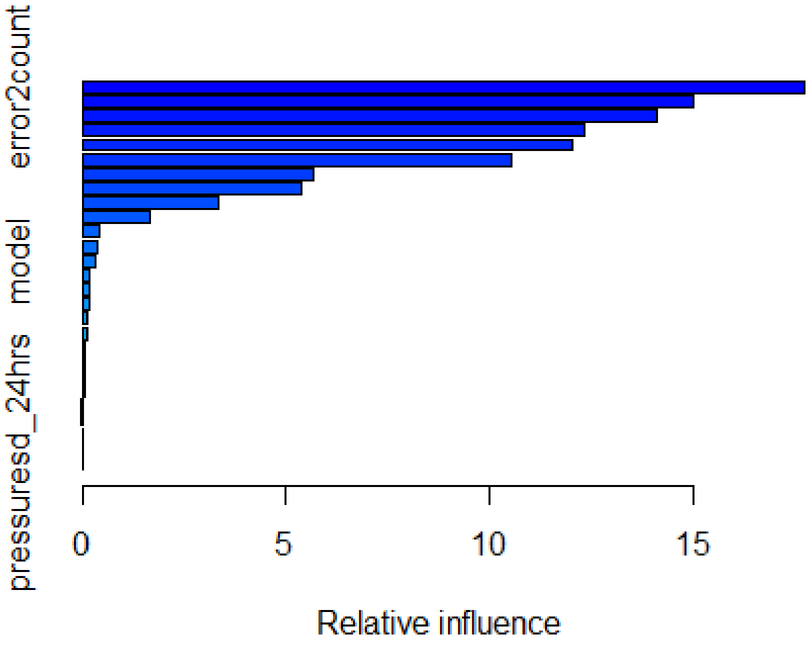
```
#print the relative influence of variables for the three models
summary(gbm_model1)
```



```
##                                      var      rel.inf
## voltmean_24hrs           voltmean_24hrs 15.800748598
## error2count                 error2count 15.451917441
## error5count                 error5count 13.558220398
## vibrationmean_24hrs vibrationmean_24hrs 12.009946044
## pressuremean_24hrs   pressuremean_24hrs 11.399599203
## error3count                 error3count 10.306045232
## rotatemean_24hrs       rotatemean_24hrs  8.175658694
## error1count                      error1count  5.859377415
## error4count                      error4count  3.500904109
## sincelastcomp1                sincelastcomp1  1.790228570
## sincelastcomp3                sincelastcomp3  0.404303231
## sincelastcomp4                sincelastcomp4  0.365227227
## model                                  model  0.341006252
## rotatemean                        rotatemean  0.175976534
## vibrationmean                  vibrationmean  0.166408027
## sincelastcomp2                sincelastcomp2  0.135260244
## pressuremean                    pressuremean  0.102963604
## pressuresd_24hrs            pressuresd_24hrs  0.092400214
## rotatesd_24hrs                rotatesd_24hrs  0.068446542
## voltmean                            voltmean  0.066363807
## rotatesd                            rotatesd  0.058408239
## pressuresd                        pressuresd  0.054289356
## vibrationsd_24hrs          vibrationsd_24hrs  0.049023260
## vibrationsd                      vibrationsd  0.033659235
## voltsd                                voltsd  0.018058874
## voltsd_24hrs                    voltsd_24hrs  0.013662522
## age                                      age  0.001897128

summary(gbm_model2)
```



```
##                                          var     rel.inf
## voltmean_24hrs                voltmean_24hrs 15.46968069
## error2count                      error2count 13.98598970
## error5count                      error5count 13.94143008
## vibrationmean_24hrs vibrationmean_24hrs 12.13422369
## pressuremean_24hrs   pressuremean_24hrs 10.86946498
## error3count                 error3count 10.31701450
## rotatemean_24hrs       rotatemean_24hrs  9.13200018
## error1count                 error1count  5.99586393
## error4count                 error4count  4.24809947
## sincelastcomp1           sincelastcomp1  1.80786380
## model                             model  0.39248119
## sincelastcomp3           sincelastcomp3  0.38182998
## sincelastcomp4           sincelastcomp4  0.31810400
## sincelastcomp2           sincelastcomp2  0.18012498
## pressuremean               pressuremean  0.16583390
## rotatemean                   rotatemean  0.14132099
## vibrationmean             vibrationmean  0.12692504
## voltmean                       voltmean  0.08473189
## pressuresd                   pressuresd  0.06444102
## rotatesd_24hrs           rotatesd_24hrs  0.03491397
## age                                 age  0.03417518
## pressuresd_24hrs       pressuresd_24hrs  0.03359059
## voltsd                           voltsd  0.03338299
## rotatesd                       rotatesd  0.03315434
## voltsd_24hrs               voltsd_24hrs  0.02796652
## vibrationsd                 vibrationsd  0.02304701
## vibrationsd_24hrs     vibrationsd_24hrs  0.02234540

summary(gbm_model3)
```



```
##                                     var     rel.inf
## error2count                 error2count 17.76680609
## voltmean_24hrs           voltmean_24hrs 15.03425740
## error5count                 error5count 14.12438122
## vibrationmean_24hrs vibrationmean_24hrs 12.32537070
## pressuremean_24hrs   pressuremean_24hrs 12.06366953
## error3count                 error3count 10.53390183
## error1count                 error1count  5.69738028
## rotatemean_24hrs       rotatemean_24hrs  5.37055553
## error4count                 error4count  3.32078036
## sincelastcomp1           sincelastcomp1  1.64559409
## sincelastcomp4           sincelastcomp4  0.43011169
## sincelastcomp3           sincelastcomp3  0.37471532
## model                             model  0.31557236
## vibrationmean             vibrationmean  0.16851993
## sincelastcomp2           sincelastcomp2  0.16321654
## rotatemean                   rotatemean  0.14032834
## pressuremean               pressuremean  0.12638741
## voltmean                       voltmean  0.10594823
## vibrationsd_24hrs     vibrationsd_24hrs  0.05892002
## rotatesd_24hrs           rotatesd_24hrs  0.04319944
## vibrationsd                 vibrationsd  0.04172146
## rotatesd                       rotatesd  0.03837903
## voltsd                           voltsd  0.03164961
## voltsd_24hrs               voltsd_24hrs  0.02596589
## age                                 age  0.01938240
## pressuresd                   pressuresd  0.01692004
## pressuresd_24hrs       pressuresd_24hrs  0.01636528
```

```
#Prediction for the first split
head(testingdata1)
```

```
##                 datetime machineID voltmean rotatemean pressuremean
## 1688 2015-08-01 02:00:00         1 157.9068   436.2231     99.66871
## 1689 2015-08-01 05:00:00         1 177.4843   474.3847     95.46521
## 1690 2015-08-01 08:00:00         1 160.7222   454.1410     96.23953
## 1691 2015-08-01 11:00:00         1 164.3274   483.3435     93.76695
## 1692 2015-08-01 14:00:00         1 168.1143   459.9587    100.90430
## 1693 2015-08-01 17:00:00         1 165.8379   452.0577     86.44224
##      vibrationmean   voltsd rotatesd pressuresd vibrationsd voltmean_24hrs
## 1688      41.92773 14.31169 48.89192   8.484343   5.7500704       164.3721
## 1689      36.53662 11.46707 52.26395   7.914226   5.9288300       164.8737
## 1690      37.22739 11.72006 60.89366  16.558994   0.5107924       165.2955
## 1691      37.09941 11.15896 36.37714   5.826244   5.4431069       165.3690
## 1692      40.96688 21.07944 70.96681   8.388908   4.2483844       165.9593
## 1693      43.77300 25.68208 57.74310   4.640864   3.7393760       165.5051
##      rotatemean_24hrs pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs
## 1688         439.8292           99.12663            39.74213     12.74752
## 1689         441.2809           98.80552            39.28227     13.48400
## 1690         450.7080           99.52644            38.99189     13.17035
## 1691         456.5083           99.38223            38.63787     11.31724
## 1692         458.8000           99.29439            39.14403     12.35963
## 1693         451.8262           97.32292            39.58123     14.39692
##      rotatesd_24hrs pressuresd_24hrs vibrationsd_24hrs error1count
error2count
## 1688       57.90303         8.341975          5.797217           0
0
## 1689       60.54160         8.660988          6.015503           0
0
## 1690       57.22895         8.868280          5.579957           0
0
## 1691       53.98503         8.969143          5.072600           0
0
## 1692       55.59771         8.665082          5.065621           0
0
## 1693       50.78838         9.506115          5.371039           0
0
##      error3count error4count error5count sincelastcomp1 sincelastcomp2
## 1688           0           0           0       12.83333       27.83333
## 1689           0           0           0       12.95833       27.95833
## 1690           0           0           0       13.08333       28.08333
## 1691           0           0           0       13.20833       28.20833
## 1692           0           0           0       13.33333       28.33333
## 1693           0           0           0       13.45833       28.45833
##      sincelastcomp3 sincelastcomp4  model age failure
## 1688       57.83333       42.83333 model3  18    none
## 1689       57.95833       42.95833 model3  18    none
## 1690       58.08333       43.08333 model3  18    none
## 1691       58.20833       43.20833 model3  18    none
## 1692       58.33333       43.33333 model3  18    none
## 1693       58.45833       43.45833 model3  18    none
```

```
pred_gbm1 <- as.data.frame(predict(gbm_model1, testingdata1, n.trees =
50,type = "response"))
names(pred_gbm1) <- gsub(".50", "", names(pred_gbm1))
pred_gbm1$failure <- as.factor(colnames(pred_gbm1)[max.col(pred_gbm1)])
head(pred_gbm1)
```

```
##          comp1        comp2        comp3        comp4      none failure
## 1 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
## 2 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
## 3 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
## 4 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
## 5 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
## 6 5.791623e-05 6.817201e-05 5.583692e-05 5.568384e-05 0.9997624    none
```

```
prediction1<-testingdata1 %>%
  mutate(failurePredicted=as.factor(pred_gbm1$failure))
head(prediction1)
```

```
##                   datetime machineID voltmean rotatemean pressuremean
## 1688 2015-08-01 02:00:00         1 157.9068   436.2231     99.66871
## 1689 2015-08-01 05:00:00         1 177.4843   474.3847     95.46521
## 1690 2015-08-01 08:00:00         1 160.7222   454.1410     96.23953
## 1691 2015-08-01 11:00:00         1 164.3274   483.3435     93.76695
## 1692 2015-08-01 14:00:00         1 168.1143   459.9587    100.90430
## 1693 2015-08-01 17:00:00         1 165.8379   452.0577     86.44224
##      vibrationmean   voltsd rotatesd pressuresd vibrationsd voltmean_24hrs
## 1688      41.92773 14.31169 48.89192   8.484343   5.7500704       164.3721
## 1689      36.53662 11.46707 52.26395   7.914226   5.9288300       164.8737
## 1690      37.22739 11.72006 60.89366  16.558994   0.5107924       165.2955
## 1691      37.09941 11.15896 36.37714   5.826244   5.4431069       165.3690
## 1692      40.96688 21.07944 70.96681   8.388908   4.2483844       165.9593
## 1693      43.77300 25.68208 57.74310   4.640864   3.7393760       165.5051
##      rotatemean_24hrs pressuremean_24hrs vibrationmean_24hrs voltsd_24hrs
## 1688         439.8292           99.12663            39.74213     12.74752
## 1689         441.2809           98.80552            39.28227     13.48400
## 1690         450.7080           99.52644            38.99189     13.17035
## 1691         456.5083           99.38223            38.63787     11.31724
## 1692         458.8000           99.29439            39.14403     12.35963
## 1693         451.8262           97.32292            39.58123     14.39692
##      rotatesd_24hrs pressuresd_24hrs vibrationsd_24hrs error1count
error2count
## 1688       57.90303         8.341975          5.797217           0
0
## 1689       60.54160         8.660988          6.015503           0
0
## 1690       57.22895         8.868280          5.579957           0
0
## 1691       53.98503         8.969143          5.072600           0
0
## 1692       55.59771         8.665082          5.065621           0
0
## 1693       50.78838         9.506115          5.371039           0
0
##      error3count error4count error5count sincelastcomp1 sincelastcomp2
## 1688           0           0           0       12.83333       27.83333
## 1689           0           0           0       12.95833       27.95833
## 1690           0           0           0       13.08333       28.08333
## 1691           0           0           0       13.20833       28.20833
## 1692           0           0           0       13.33333       28.33333
## 1693           0           0           0       13.45833       28.45833
##      sincelastcomp3 sincelastcomp4  model age failure failurePredicted
## 1688       57.83333       42.83333 model3  18    none             none
## 1689       57.95833       42.95833 model3  18    none             none
## 1690       58.08333       43.08333 model3  18    none             none
## 1691       58.20833       43.20833 model3  18    none             none
## 1692       58.33333       43.33333 model3  18    none             none
## 1693       58.45833       43.45833 model3  18    none             none
```

```
#we can analyse the errors in the prediction as done in the following

#FIRST ANALYSIS
#we can analyse the entire set of predictions of "none" state
#we can limit the analysis to the datetime, failure and failurePredicted
columns

prediction_analysis<-prediction1 %>%
                             filter(failure=="none" &&
failurePredicted!="none") %>%
                             select(datetime, machineID, failure,
failurePredicted)

## Warning in failure == "none" && failurePredicted != "none": 'length(x) =
122752
## > 1' in coercion to 'logical(1)'

## Warning in failure == "none" && failurePredicted != "none": 'length(x) =
122752
## > 1' in coercion to 'logical(1)'

head(prediction_analysis)

## [1] datetime         machineID        failure          failurePredicted
## <0 rows> (or 0-length row.names)

#SECOND ANALYSIS
#we can analyse the entire set of failure predictions (without "none" state)
#we can limit the analysis to the datetime, failure and failurePredicted
columns

prediction_analysis<-prediction1 %>%
                             filter(failure!="none") %>%
                             select(datetime, machineID, failure,
failurePredicted)
head(prediction_analysis)

##              datetime machineID failure failurePredicted
## 1 2015-09-01 08:00:00         1   comp4            comp4
## 2 2015-09-01 11:00:00         1   comp4            comp4
## 3 2015-09-01 14:00:00         1   comp4            comp4
## 4 2015-09-01 17:00:00         1   comp4            comp4
## 5 2015-09-01 20:00:00         1   comp4            comp4
## 6 2015-09-01 23:00:00         1   comp4            comp4

#we can analyse the wrong predictions
#we can limit the analysis to the datetime, failure and failurePredicted
columns
prediction_analysis=filter(prediction1, failure!=failurePredicted)
prediction_analysis=select(prediction_analysis, datetime, machineID, failure,
failurePredicted)
head(prediction_analysis)
```

```
##              datetime machineID failure failurePredicted
## 1 2015-10-17 08:00:00         8   comp4             none
## 2 2015-09-04 02:00:00        12   comp1             none
## 3 2015-08-23 08:00:00        13   comp3            comp1
## 4 2015-08-23 11:00:00        13   comp3            comp1
## 5 2015-08-23 14:00:00        13   comp3            comp1
## 6 2015-08-23 17:00:00        13   comp3            comp1
```

```
#FINAL STEP: EVALUATION

# define evaluate function
Evaluate<-function(actual=NULL, predicted=NULL, cm=NULL){
  if(is.null(cm)) {
    actual = actual[!is.na(actual)]
    predicted = predicted[!is.na(predicted)]
    f = factor(union(unique(actual), unique(predicted)))
    actual = factor(actual, levels = levels(f))
    predicted = factor(predicted, levels = levels(f))
    cm = as.matrix(table(Actual=actual, Predicted=predicted))
  }
  n = sum(cm) # number of instances
  nc = nrow(cm) # number of classes
  diag = diag(cm) # number of correctly classified instances per class
  rowsums = apply(cm, 1, sum) # number of instances per class
  colsums = apply(cm, 2, sum) # number of predictions per class
  p = rowsums / n # distribution of instances over the classes
  q = colsums / n # distribution of instances over the predicted classes
  #accuracy
  accuracy = sum(diag) / n
  #per class
  recall = diag / rowsums
  precision = diag / colsums
  f1 = 2 * precision * recall / (precision + recall)
  #macro
  macroPrecision = mean(precision)
  macroRecall = mean(recall)
  macroF1 = mean(f1)
  #1-vs-all matrix
  oneVsAll = lapply(1 : nc,
                    function(i){
                      v = c(cm[i,i],
                            rowsums[i] - cm[i,i],
                            colsums[i] - cm[i,i],
                            n-rowsums[i] - colsums[i] + cm[i,i]);
                      return(matrix(v, nrow = 2, byrow = T))})
  s = matrix(0, nrow=2, ncol=2)
  for(i in 1:nc){s=s+oneVsAll[[i]]}
  #avg accuracy
  avgAccuracy = sum(diag(s))/sum(s)
  #micro
  microPrf = (diag(s) / apply(s,1, sum))[1];
  #majority class
  mcIndex = which(rowsums==max(rowsums))[1] # majority-class index
  mcAccuracy = as.numeric(p[mcIndex])
  mcRecall = 0*p; mcRecall[mcIndex] = 1
  mcPrecision = 0*p; mcPrecision[mcIndex] = p[mcIndex]
  mcF1 = 0*p; mcF1[mcIndex] = 2 * mcPrecision[mcIndex] /
(mcPrecision[mcIndex] + 1)
  #random accuracy
  expAccuracy = sum(p*q)
  #kappa
  kappa = (accuracy - expAccuracy) / (1 - expAccuracy)
  #random guess
  rgAccuracy = 1 / nc
  rgPrecision = p
  rgRecall = 0*p + 1 / nc
  rgF1 = 2 * p / (nc * p + 1)
  #rnd weighted
  rwgAccurcy = sum(p^2)
  rwgPrecision = p
  rwgRecall = p
  rwgF1 = p
  classNames = names(diag)
  if(is.null(classNames)) classNames = paste("C",(1:nc),sep="")
  return(list(
    ConfusionMatrix = cm,
    Metrics = data.frame(
      Class = classNames,
      Accuracy = accuracy,
      Precision = precision,
      Recall = recall,
      F1 = f1,
      MacroAvgPrecision = macroPrecision,
      MacroAvgRecall = macroRecall,
      MacroAvgF1 = macroF1,
      AvgAccuracy = avgAccuracy,
      MicroAvgPrecision = microPrf,
      MicroAvgRecall = microPrf,
      MicroAvgF1 = microPrf,
      MajorityClassAccuracy = mcAccuracy,
      MajorityClassPrecision = mcPrecision,
      MajorityClassRecall = mcRecall,
      MajorityClassF1 = mcF1,
      Kappa = kappa,
      RandomGuessAccuracy = rgAccuracy,
      RandomGuessPrecision = rgPrecision,
      RandomGuessRecall = rgRecall,
      RandomGuessF1 = rgF1,
      RandomWeightedGuessAccurcy = rwgAccurcy,
      RandomWeightedGuessPrecision = rwgPrecision,
      RandomWeightedGuessRecall= rwgRecall,
      RandomWeightedGuessWeightedF1 = rwgF1)))
}

# evaluation metrics for first split
pred_gbm1 <- as.data.frame(predict(gbm_model1, testingdata1, n.trees =
50,type = "response"))
names(pred_gbm1) <- gsub(".50", "", names(pred_gbm1))
pred_gbm1$failure <- as.factor(colnames(pred_gbm1)[max.col(pred_gbm1)])
eval1 <- Evaluate(actual=testingdata1$failure,predicted=pred_gbm1$failure)
eval1$ConfusionMatrix
```

```
##         Predicted
## Actual   comp1  comp2  comp3  comp4   none
##   comp1    498     15      0      8      7
##   comp2      9    834      4     41      2
##   comp3     24     12    378      2      0
##   comp4     16     15      6    546      1
##   none      15      0      0      5 120314
```

```
t(eval1$Metrics)
```

```
##                                 comp1         comp2         comp3
## Class                           "comp1"       "comp2"       "comp3"
## Accuracy                        "0.9985173"   "0.9985173"   "0.9985173"
## Precision                       "0.8861210"   "0.9520548"   "0.9742268"
## Recall                          "0.9431818"   "0.9370787"   "0.9086538"
## F1                              "0.9137615"   "0.9445074"   "0.9402985"
## MacroAvgPrecision               "0.9438592"   "0.9438592"   "0.9438592"
## MacroAvgRecall                  "0.9447359"   "0.9447359"   "0.9447359"
## MacroAvgF1                      "0.9438369"   "0.9438369"   "0.9438369"
## AvgAccuracy                     "0.9994069"   "0.9994069"   "0.9994069"
## MicroAvgPrecision               "0.9985173"   "0.9985173"   "0.9985173"
## MicroAvgRecall                  "0.9985173"   "0.9985173"   "0.9985173"
## MicroAvgF1                      "0.9985173"   "0.9985173"   "0.9985173"
## MajorityClassAccuracy           "0.9803017"   "0.9803017"   "0.9803017"
## MajorityClassPrecision          "0.0000000"   "0.0000000"   "0.0000000"
## MajorityClassRecall             "0"           "0"           "0"
## MajorityClassF1                 "0.0000000"   "0.0000000"   "0.0000000"
## Kappa                           "0.9619663"   "0.9619663"   "0.9619663"
## RandomGuessAccuracy             "0.2"         "0.2"         "0.2"
## RandomGuessPrecision            "0.004301356" "0.007250391" "0.003388947"
## RandomGuessRecall               "0.2"         "0.2"         "0.2"
## RandomGuessF1                   "0.008421590" "0.013993491" "0.006664958"
## RandomWeightedGuessAccurcy      "0.9610967"   "0.9610967"   "0.9610967"
## RandomWeightedGuessPrecision    "0.004301356" "0.007250391" "0.003388947"
## RandomWeightedGuessRecall       "0.004301356" "0.007250391" "0.003388947"
## RandomWeightedGuessWeightedF1   "0.004301356" "0.007250391" "0.003388947"
##                               comp4         none
## Class                         "comp4"       "none"
## Accuracy                      "0.9985173"   "0.9985173"
## Precision                     "0.9069767"   "0.9999169"
## Recall                        "0.9349315"   "0.9998338"
## F1                            "0.9207420"   "0.9998753"
## MacroAvgPrecision             "0.9438592"   "0.9438592"
## MacroAvgRecall                "0.9447359"   "0.9447359"
## MacroAvgF1                    "0.9438369"   "0.9438369"
## AvgAccuracy                   "0.9994069"   "0.9994069"
## MicroAvgPrecision             "0.9985173"   "0.9985173"
## MicroAvgRecall                "0.9985173"   "0.9985173"
## MicroAvgF1                    "0.9985173"   "0.9985173"
## MajorityClassAccuracy         "0.9803017"   "0.9803017"
## MajorityClassPrecision        "0.0000000"   "0.9803017"
## MajorityClassRecall           "0"           "1"
## MajorityClassF1               "0.0000000"   "0.9900529"
## Kappa                         "0.9619663"   "0.9619663"
## RandomGuessAccuracy           "0.2"         "0.2"
## RandomGuessPrecision          "0.004757560" "0.980301747"
## RandomGuessRecall             "0.2"         "0.2"
## RandomGuessF1                 "0.009294035" "0.332220722"
## RandomWeightedGuessAccurcy    "0.9610967"   "0.9610967"
## RandomWeightedGuessPrecision  "0.004757560" "0.980301747"
## RandomWeightedGuessRecall     "0.004757560" "0.980301747"
## RandomWeightedGuessWeightedF1 "0.004757560" "0.980301747"
```

```
# evaluation metrics for second split
pred_gbm2 <- as.data.frame(predict(gbm_model2, testingdata2, n.trees =
50,type = "response"))
names(pred_gbm2) <- gsub(".50", "", names(pred_gbm2))
pred_gbm2$failure <- as.factor(colnames(pred_gbm2)[max.col(pred_gbm2)])
eval2 <- Evaluate(actual=testingdata2$failure,predicted=pred_gbm2$failure)
eval2$ConfusionMatrix
```

```
##         Predicted
## Actual   comp1 comp2 comp3 comp4  none
##   comp1    379    13     1     8     7
##   comp2      3   705     9     4     1
##   comp3      7     7   303     3     0
##   comp4     16    22     5   406     1
##   none      11     0     0     5 95982
```

```
t(eval2$Metrics)
```

```
##                               comp1         comp2         comp3
## Class                         "comp1"       "comp2"       "comp3"
## Accuracy                      "0.9987436"   "0.9987436"   "0.9987436"
## Precision                     "0.9110577"   "0.9437751"   "0.9528302"
## Recall                        "0.9289216"   "0.9764543"   "0.9468750"
## F1                            "0.9199029"   "0.9598366"   "0.9498433"
## MacroAvgPrecision             "0.9521242"   "0.9521242"   "0.9521242"
## MacroAvgRecall                "0.9508613"   "0.9508613"   "0.9508613"
## MacroAvgF1                    "0.9512786"   "0.9512786"   "0.9512786"
## AvgAccuracy                   "0.9994974"   "0.9994974"   "0.9994974"
## MicroAvgPrecision             "0.9987436"   "0.9987436"   "0.9987436"
## MicroAvgRecall                "0.9987436"   "0.9987436"   "0.9987436"
## MicroAvgF1                    "0.9987436"   "0.9987436"   "0.9987436"
## MajorityClassAccuracy         "0.980592"    "0.980592"    "0.980592"
## MajorityClassPrecision        "0.000000"    "0.000000"    "0.000000"
## MajorityClassRecall           "0"           "0"           "0"
## MajorityClassF1               "0.0000000"   "0.0000000"   "0.0000000"
## Kappa                         "0.967285"    "0.967285"    "0.967285"
## RandomGuessAccuracy           "0.2"         "0.2"         "0.2"
## RandomGuessPrecision          "0.004167603" "0.007375023" "0.003268708"
## RandomGuessRecall             "0.2"         "0.2"         "0.2"
## RandomGuessF1                 "0.008165062" "0.014225480" "0.006432290"
## RandomWeightedGuessAccurcy    "0.9616643"   "0.9616643"   "0.9616643"
## RandomWeightedGuessPrecision  "0.004167603" "0.007375023" "0.003268708"
## RandomWeightedGuessRecall     "0.004167603" "0.007375023" "0.003268708"
## RandomWeightedGuessWeightedF1 "0.004167603" "0.007375023" "0.003268708"
##                               comp4         none
## Class                         "comp4"       "none"
## Accuracy                      "0.9987436"   "0.9987436"
## Precision                     "0.9530516"   "0.9999062"
## Recall                        "0.9022222"   "0.9998333"
## F1                            "0.9269406"   "0.9998698"
## MacroAvgPrecision             "0.9521242"   "0.9521242"
## MacroAvgRecall                "0.9508613"   "0.9508613"
## MacroAvgF1                    "0.9512786"   "0.9512786"
## AvgAccuracy                   "0.9994974"   "0.9994974"
## MicroAvgPrecision             "0.9987436"   "0.9987436"
## MicroAvgRecall                "0.9987436"   "0.9987436"
## MicroAvgF1                    "0.9987436"   "0.9987436"
## MajorityClassAccuracy         "0.980592"    "0.980592"
## MajorityClassPrecision        "0.000000"    "0.980592"
## MajorityClassRecall           "0"           "1"
## MajorityClassF1               "0.0000000"   "0.9902009"
## Kappa                         "0.967285"    "0.967285"
## RandomGuessAccuracy           "0.2"         "0.2"
## RandomGuessPrecision          "0.004596621" "0.980592045"
## RandomGuessRecall             "0.2"         "0.2"
## RandomGuessF1                 "0.008986700" "0.332237389"
## RandomWeightedGuessAccurcy    "0.9616643"   "0.9616643"
## RandomWeightedGuessPrecision  "0.004596621" "0.980592045"
## RandomWeightedGuessRecall     "0.004596621" "0.980592045"
## RandomWeightedGuessWeightedF1 "0.004596621" "0.980592045"

# evaluation metrics for third split
pred_gbm3 <- as.data.frame(predict(gbm_model3, testingdata3, n.trees =
50,type = "response"))
```

```
names(pred_gbm3)<-gsub(".50", "", names(pred_gbm3))
pred_gbm3$failure <- as.factor(colnames(pred_gbm3)[max.col(pred_gbm3)])
eval3 <- Evaluate(actual=testingdata3$failure,predicted=pred_gbm3$failure)
eval3$ConfusionMatrix
```

```
##        Predicted
## Actual  comp1 comp2 comp3 comp4  none
##   comp1   284    15     0     5     2
##   comp2     1   557     0    10     2
##   comp3     8     0   212     4     0
##   comp4    11    14     4   292     1
##   none      7     0     0     5 72438
```

```
t(eval3$Metrics)
```

```
##                                 comp1         comp2         comp3        
## Class                           "comp1"       "comp2"       "comp3"      
## Accuracy                        "0.9987952"   "0.9987952"   "0.9987952"  
## Precision                       "0.9131833"   "0.9505119"   "0.9814815"  
## Recall                          "0.9281046"   "0.9771930"   "0.9464286"  
## F1                              "0.9205835"   "0.9636678"   "0.9636364"  
## MacroAvgPrecision               "0.9538317"   "0.9538317"   "0.9538317"  
## MacroAvgRecall                  "0.9516786"   "0.9516786"   "0.9516786"  
## MacroAvgF1                      "0.9526262"   "0.9526262"   "0.9526262"  
## AvgAccuracy                     "0.9995181"   "0.9995181"   "0.9995181"  
## MicroAvgPrecision               "0.9987952"   "0.9987952"   "0.9987952"  
## MicroAvgRecall                  "0.9987952"   "0.9987952"   "0.9987952"  
## MicroAvgF1                      "0.9987952"   "0.9987952"   "0.9987952"  
## MajorityClassAccuracy           "0.9807505"   "0.9807505"   "0.9807505"  
## MajorityClassPrecision          "0.0000000"   "0.0000000"   "0.0000000"  
## MajorityClassRecall             "0"           "0"           "0"          
## MajorityClassF1                 "0.0000000"   "0.0000000"   "0.0000000"  
## Kappa                           "0.968391"    "0.968391"    "0.968391"   
## RandomGuessAccuracy             "0.2"         "0.2"         "0.2"        
## RandomGuessPrecision            "0.004142300" "0.007716049" "0.003032272"
## RandomGuessRecall               "0.2"         "0.2"         "0.2"        
## RandomGuessF1                   "0.008116496" "0.014858841" "0.005973971"
## RandomWeightedGuessAccurcy      "0.9619764"   "0.9619764"   "0.9619764"  
## RandomWeightedGuessPrecision    "0.004142300" "0.007716049" "0.003032272"
## RandomWeightedGuessRecall       "0.004142300" "0.007716049" "0.003032272"
## RandomWeightedGuessWeightedF1   "0.004142300" "0.007716049" "0.003032272"
##                                 comp4         none         
## Class                           "comp4"       "none"       
## Accuracy                        "0.9987952"   "0.9987952"  
## Precision                       "0.9240506"   "0.9999310"  
## Recall                          "0.9068323"   "0.9998344"  
## F1                              "0.9153605"   "0.9998827"  
## MacroAvgPrecision               "0.9538317"   "0.9538317"  
## MacroAvgRecall                  "0.9516786"   "0.9516786"  
## MacroAvgF1                      "0.9526262"   "0.9526262"  
## AvgAccuracy                   "0.9995181"   "0.9995181"
## MicroAvgPrecision             "0.9987952"   "0.9987952"
## MicroAvgRecall                "0.9987952"   "0.9987952"
## MicroAvgF1                    "0.9987952"   "0.9987952"
## MajorityClassAccuracy         "0.9807505"   "0.9807505"
## MajorityClassPrecision        "0.0000000"   "0.9807505"
## MajorityClassRecall           "0"           "1"
## MajorityClassF1               "0.0000000"   "0.9902817"
## Kappa                         "0.968391"    "0.968391"
## RandomGuessAccuracy           "0.2"         "0.2"
## RandomGuessPrecision          "0.004358891" "0.980750487"
## RandomGuessRecall             "0.2"         "0.2"
## RandomGuessF1                 "0.008531835" "0.332246481"
## RandomWeightedGuessAccurcy    "0.9619764"   "0.9619764"
## RandomWeightedGuessPrecision  "0.004358891" "0.980750487"
## RandomWeightedGuessRecall     "0.004358891" "0.980750487"
## RandomWeightedGuessWeightedF1 "0.004358891" "0.980750487"
```

```
# report the RECALL rates for the models
rownames <- c("comp1","comp2","comp3","comp4","none")
data.frame(cbind(failure = rownames,
                 gbm_model1_Recall = eval1$Metrics$Recall,
                 gbm_model2_Recall = eval2$Metrics$Recall,
                 gbm_model3_Recall = eval3$Metrics$Recall))
```

```
##   failure gbm_model1_Recall gbm_model2_Recall gbm_model3_Recall
## 1   comp1 0.943181818181818 0.928921568627451 0.928104575163399
## 2   comp2 0.937078651685393 0.976454293628809  0.97719298245614
## 3   comp3 0.908653846153846          0.946875 0.946428571428571
## 4   comp4 0.934931506849315 0.902222222222222 0.906832298136646
## 5    none 0.999833795934649 0.999833329861039 0.999834368530021
```