# Mantenimiento Preventivo de Máquinas

## Machine Learning Predictivo

15/2/2022



---

# 1 ID Petición

- Número 3: Predicción del fallo en una máquina para realización de Plan de Mantenimiento Preventivo.

# 2 Contexto

- El equipo de dirección quiere tener un Plan de Mantenimiento Preventivo de la máquinaria con la que cuenta la empresa.

- Realizar el mantenimiento preventivo de la maquinaria evita averías, sustitución de piezas no fungibles, además de cortes en la producción e incremento en el coste de mano de obra.

- El departamento de mantenimiento quiere un plan concreto de mantenimiento preventivo de la maquinaria de la empresa, para organización de las tareas del personal del departamento.

# 3 Requirimientos del proyecto

- Realización del estudio de probabilidad de fallo en la maquinaria.

- Optimizar las labores del personal de mantenimiento.

---

# 4 Detalle del proceso

## 4.1 - Preparación del entorno

- 4.1.1 - Cargamos las librerías que vamos a utilizar

```
paquetes <- c('dplyr',
              'skimr',
              'lubridate',
              'tidyr',
              'ggplot2',
              'ROCR')

instalados <- paquetes %in% installed.packages()

if(sum(instalados == FALSE) > 0) {
  install.packages(paquetes[!instalados])
}
lapply(paquetes,require,character.only = TRUE)
```

```
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
##
## [[6]]
## [1] TRUE
```

Parámetros - Desactivamos la notación científica:

```
options(scipen=999)
```

- 4.1.2 - Cargamos los datos

Procedentes de la base de datos histórica de la compañía, archivo '.csv'.

```
df <- read.csv('MntoMaquina.csv')
```

# 4.2 - Análisis exploratorio

- 4.2.1 - Análisis exploratorio general y tipo de datos

Vemos las variables con las que trabajamos y el tipo de dato de cada una.

```
glimpse(df)
```

```
## Rows: 7,027
## Columns: 28
## $ ï..Date                      <chr> "2016-01-01 01:00:00", "2016-01-01 02:00:~
## $ Temperature                  <int> 68, 64, 63, 65, 67, 65, 63, 61, 62, 62, 6~
## $ Humidity                     <int> 77, 76, 80, 81, 76, 80, 80, 83, 81, 76, 8~
## $ Operator                     <chr> "Operator1", "Operator1", "Operator1", "O~
## $ Measure1                     <int> 1180, 1406, 550, 1928, 1021, 1731, 415, 5~
## $ Measure2                     <int> 1, 1, 1, 1, 2, 2, 0, 2, 3, 0, 1, 3, 0, 3,~
## $ Measure3                     <int> 1, 1, 1, 2, 1, 0, 0, 2, 1, 0, 1, 0, 2, 2,~
## $ Measure4                     <int> 1915, 511, 1754, 1326, 185, 1424, 1008, 6~
## $ Measure5                     <int> 1194, 1577, 1834, 1082, 170, 1176, 1086, ~
## $ Measure6                     <int> 637, 1121, 1413, 233, 952, 1223, 1759, 17~
## $ Measure7                     <int> 1093, 1948, 1151, 1441, 1183, 621, 1946, ~
## $ Measure8                     <int> 524, 1882, 945, 1736, 1329, 647, 1814, 64~
## $ Measure9                     <int> 919, 1301, 1312, 1033, 427, 369, 1754, 31~
## $ Measure10                    <int> 245, 273, 1494, 1549, 1638, 239, 1442, 93~
## $ Measure11                    <int> 403, 1927, 1755, 802, 850, 1196, 341, 189~
## $ Measure12                    <int> 723, 1123, 1434, 1819, 379, 1944, 1097, 1~
## $ Measure13                    <int> 1446, 717, 502, 1616, 1529, 1583, 1819, 1~
## $ Measure14                    <int> 719, 1518, 1336, 1507, 755, 1630, 472, 15~
## $ Measure15                    <int> 748, 1689, 711, 507, 844, 237, 491, 1102,~
## $ Hours.Since.Previous.Failure <int> 91, 92, 93, 94, 97, 98, 99, 100, 101, 102~
## $ Failure                      <chr> "No", "No", "No", "No", "No", "No", "No",~
## $ Date.year                    <int> 2016, 2016, 2016, 2016, 2016, 2016, 2016,~
## $ Date.month                   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ Date.day.of.month            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ Date.day.of.week             <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,~
## $ Date.hour                    <int> 1, 2, 3, 4, 7, 8, 9, 10, 11, 12, 13, 14, ~
## $ Date.minute                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ Date.second                  <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

Tenemos un conjunto de datos de 28 columnas y 7027 filas.

Esta función nos ofrece unos pequeños gráficos para ver el perfil de las variables, así como los datos faltantes, la media, los máximos y mínimos y los percentiles por cada variable.

```
skim(df)
```

Data summary

| | |
|---|---|
| Name | df |
| Number of rows | 7027 |
| Number of columns | 28 |
| ________________________ | |
| Column type frequency: | |
| character | 3 |
| numeric | 25 |
| ________________________ | |
| Group variables | None |

**Variable type: character**

| skim_variable | n_missing | complete_rate | min | max | empty | n_unique | whitespace |
|---|---|---|---|---|---|---|---|
| ï..Date | 0 | 1 | 19 | 19 | 0 | 7027 | 0 |
| Operator | 0 | 1 | 9 | 9 | 0 | 8 | 0 |

| skim_variable | n_missing | complete_rate | min | max | empty | n_unique | whitespace |
|---|---|---|---|---|---|---|---|
| Failure | 0 | 1 | 2 | 3 | 0 | 2 | 0 |

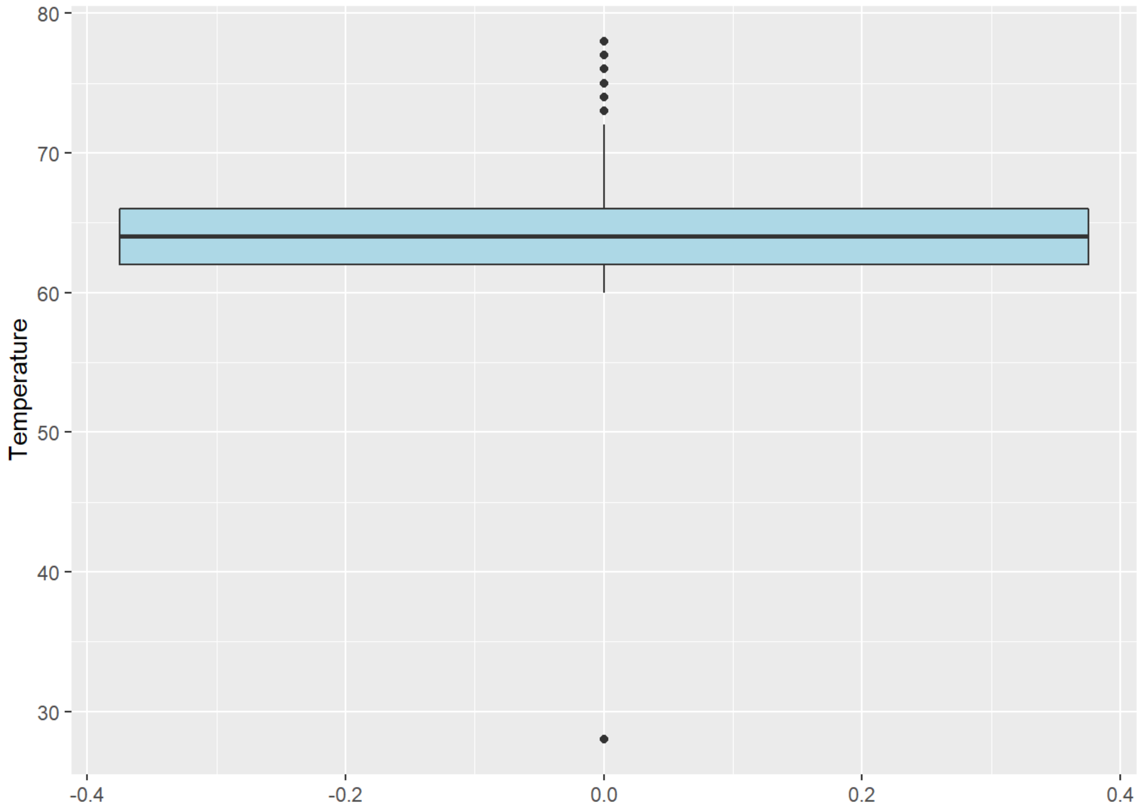
**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
|---|---|---|---|---|---|---|---|---|---|---|
| Temperature | 0 | 1 | 64.03 | 2.70 | 28 | 62.0 | 64 | 66.0 | 78 | |
| Humidity | 0 | 1 | 83.33 | 4.85 | 65 | 80.0 | 83 | 87.0 | 122 | |
| Measure1 | 0 | 1 | 1090.76 | 536.00 | 155 | 631.0 | 1090 | 1555.0 | 2011 | |
| Measure2 | 0 | 1 | 1.50 | 1.12 | 0 | 0.0 | 2 | 2.0 | 3 | |
| Measure3 | 0 | 1 | 1.01 | 0.82 | 0 | 0.0 | 1 | 2.0 | 2 | |
| Measure4 | 0 | 1 | 1077.52 | 537.16 | 155 | 617.0 | 1069 | 1541.0 | 2011 | |
| Measure5 | 0 | 1 | 1067.81 | 531.94 | 155 | 596.0 | 1070 | 1531.0 | 2011 | |
| Measure6 | 0 | 1 | 1075.19 | 534.40 | 155 | 622.0 | 1072 | 1538.0 | 2011 | |
| Measure7 | 0 | 1 | 1089.31 | 539.90 | 155 | 618.5 | 1091 | 1563.0 | 2011 | |
| Measure8 | 0 | 1 | 1074.99 | 538.53 | 155 | 606.5 | 1075 | 1534.0 | 2011 | |
| Measure9 | 0 | 1 | 1080.66 | 531.36 | 155 | 629.5 | 1078 | 1525.0 | 2011 | |
| Measure10 | 0 | 1 | 1078.82 | 537.29 | 155 | 617.0 | 1073 | 1541.5 | 2011 | |
| Measure11 | 0 | 1 | 1090.72 | 535.17 | 155 | 627.0 | 1097 | 1552.5 | 2011 | |
| Measure12 | 0 | 1 | 1088.06 | 531.60 | 155 | 631.0 | 1083 | 1547.5 | 2011 | |
| Measure13 | 0 | 1 | 1076.29 | 534.08 | 155 | 605.0 | 1071 | 1540.0 | 2011 | |
| Measure14 | 0 | 1 | 1085.28 | 538.22 | 155 | 611.0 | 1087 | 1558.5 | 2011 | |
| Measure15 | 0 | 1 | 1085.99 | 537.12 | 155 | 620.0 | 1079 | 1554.0 | 2011 | |
| Hours.Since.Previous.Failure | 0 | 1 | 216.08 | 152.39 | 1 | 88.0 | 192 | 323.0 | 666 | |
| Date.year | 0 | 1 | 2016.00 | 0.00 | 2016 | 2016.0 | 2016 | 2016.0 | 2016 | |
| Date.month | 0 | 1 | 6.51 | 3.46 | 1 | 3.0 | 7 | 10.0 | 12 | |
| Date.day.of.month | 0 | 1 | 15.75 | 8.81 | 1 | 8.0 | 16 | 23.0 | 31 | |
| Date.day.of.week | 0 | 1 | 4.01 | 2.00 | 1 | 2.0 | 4 | 6.0 | 7 | |
| Date.hour | 0 | 1 | 11.47 | 6.91 | 0 | 5.0 | 11 | 17.0 | 23 | |
| Date.minute | 0 | 1 | 0.00 | 0.00 | 0 | 0.0 | 0 | 0.0 | 0 | |
| Date.second | 0 | 1 | 0.00 | 0.00 | 0 | 0.0 | 0 | 0.0 | 0 | |

De la observación de estos datos, podemos concluir: - No hay nulos - Measure2 y Measure3 parecen factores, en lugar de enteros. - Viendo el mínimo y el p25 de Temperature parece que tiene algunos datos atípicos.

Analizamos en mayor detalle la tempertura y los datos atípicos que hemos detectado.

```
ggplot(df,x=1) + geom_boxplot(aes(y=Temperature), fill='lightblue')
```




Conclusión: efectivamente vemos que hay varios valores atípicos por debajo del cuartil 25.

- 4.2.2 - Calidad de datos:

Hacemos las siguientes acciones sobre el dataframe original:

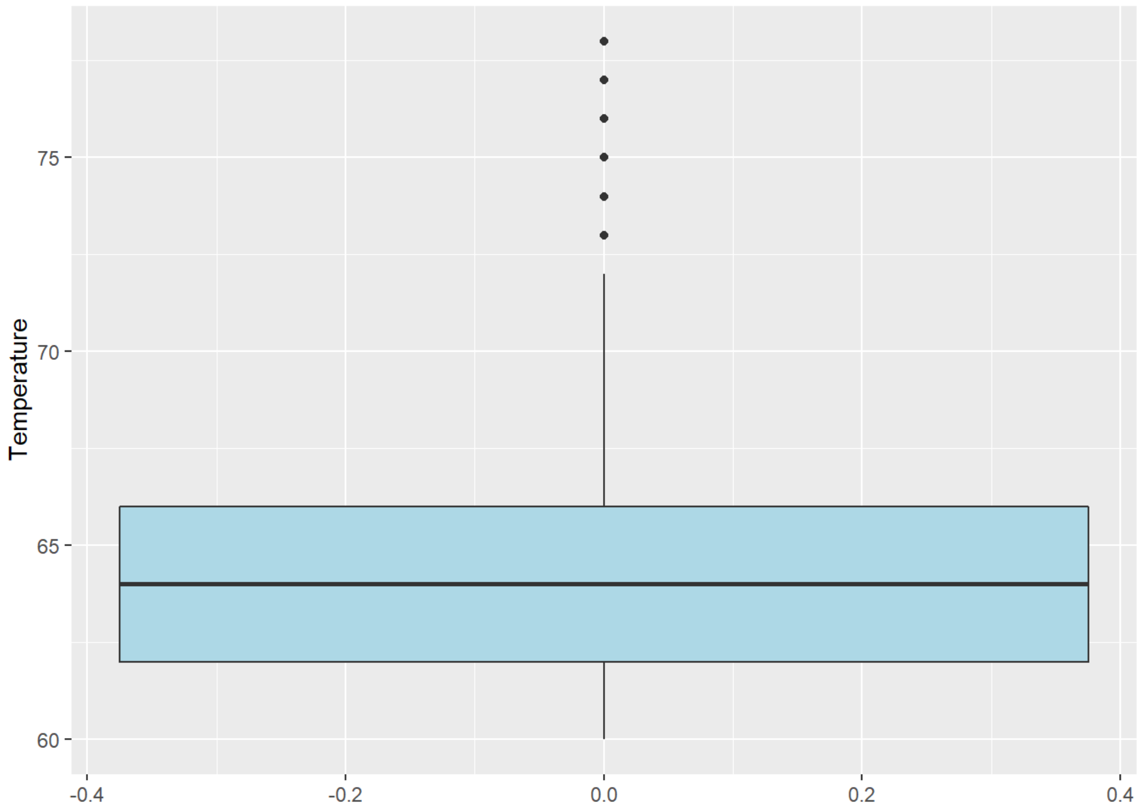
Pasamos a factor las variables Measure 2 y 3, además de la variable del Operator y la de Failure, que solo tiene valores 'Yes' y 'No'.

También eliminamos los valores de temperatura que estén por debajo de 45 grados, los cuales hemos considerado valores atípicos.

```
df <- df %>%
  mutate(Measure2 = as.factor(Measure2),
         Measure3 = as.factor(Measure3),
         Operator = as.factor(Operator),
         Failure = as.factor(Failure)) %>%
  filter(Temperature > 45)
```

Comprobamos la variable temperatura con un gráfico.

```
ggplot(df,x=1) + geom_boxplot(aes(y=Temperature), fill='lightblue')
```




Comprobamos también que Measure 2 y 3 han pasado a tipo de datos: factor.

```
glimpse(df)
```

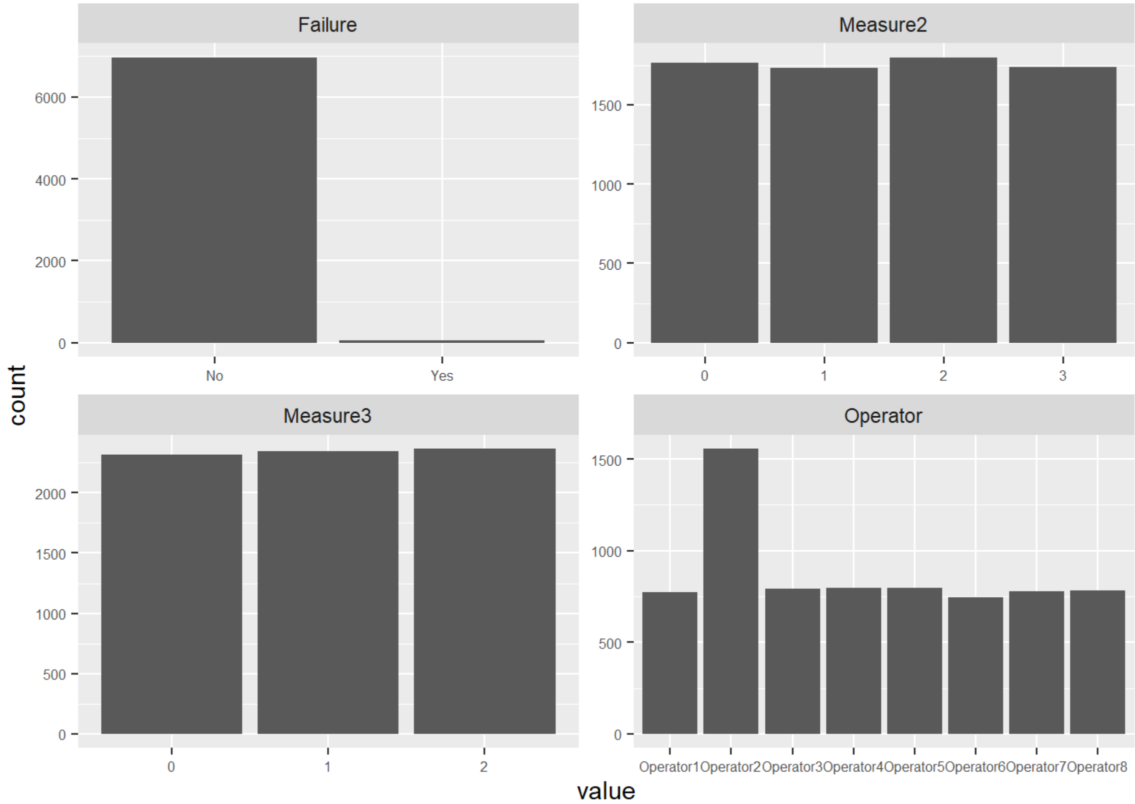
```
## Rows: 7,026
## Columns: 28
## $ ï..Date                  <chr> "2016-01-01 01:00:00", "2016-01-01 02:00:~
## $ Temperature              <int> 68, 64, 63, 65, 67, 65, 63, 61, 62, 62, 6~
## $ Humidity                 <int> 77, 76, 80, 81, 76, 80, 80, 83, 81, 76, 8~
## $ Operator                 <fct> Operator1, Operator1, Operator1, Operator~
## $ Measure1                 <int> 1180, 1406, 550, 1928, 1021, 1731, 415, 5~
## $ Measure2                 <fct> 1, 1, 1, 1, 2, 2, 0, 2, 3, 0, 1, 3, 0, 3,~
## $ Measure3                 <fct> 1, 1, 1, 2, 1, 0, 0, 2, 1, 0, 1, 0, 2, 2,~
## $ Measure4                 <int> 1915, 511, 1754, 1326, 185, 1424, 1008, 6~
## $ Measure5                 <int> 1194, 1577, 1834, 1082, 170, 1176, 1086, ~
## $ Measure6                 <int> 637, 1121, 1413, 233, 952, 1223, 1759, 17~
## $ Measure7                 <int> 1093, 1948, 1151, 1441, 1183, 621, 1946, ~
## $ Measure8                 <int> 524, 1882, 945, 1736, 1329, 647, 1814, 64~
## $ Measure9                 <int> 919, 1301, 1312, 1033, 427, 369, 1754, 31~
## $ Measure10                <int> 245, 273, 1494, 1549, 1638, 239, 1442, 93~
## $ Measure11                <int> 403, 1927, 1755, 802, 850, 1196, 341, 189~
## $ Measure12                <int> 723, 1123, 1434, 1819, 379, 1944, 1097, 1~
## $ Measure13                <int> 1446, 717, 502, 1616, 1529, 1583, 1819, 1~
## $ Measure14                <int> 719, 1518, 1336, 1507, 755, 1630, 472, 15~
## $ Measure15                <int> 748, 1689, 711, 507, 844, 237, 491, 1102,~
## $ Hours.Since.Previous.Failure <int> 91, 92, 93, 94, 97, 98, 99, 100, 101, 102~
## $ Failure                  <fct> No, No, No, No, No, No, No, No, No, No, N~
## $ Date.year                <int> 2016, 2016, 2016, 2016, 2016, 2016, 2016,~
## $ Date.month               <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ Date.day.of.month        <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ Date.day.of.week         <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,~
## $ Date.hour                <int> 1, 2, 3, 4, 7, 8, 9, 10, 11, 12, 13, 14, ~
## $ Date.minute              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ Date.second              <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

El dataframe actualmente consta de 7026 filas, se ha eliminado la fila del valor atípico de la Temperatura.

- 4.2.3 - Calidad de datos: Análisis de atípicos

4.2.3.1 - Analizamos las que son de tipo factor

```
df %>%
  select_if(is.factor) %>%
  gather() %>%
  ggplot(aes(value)) + geom_bar() + facet_wrap(~key,scales='free') + theme(axis.text=element_text
(size=6))
```

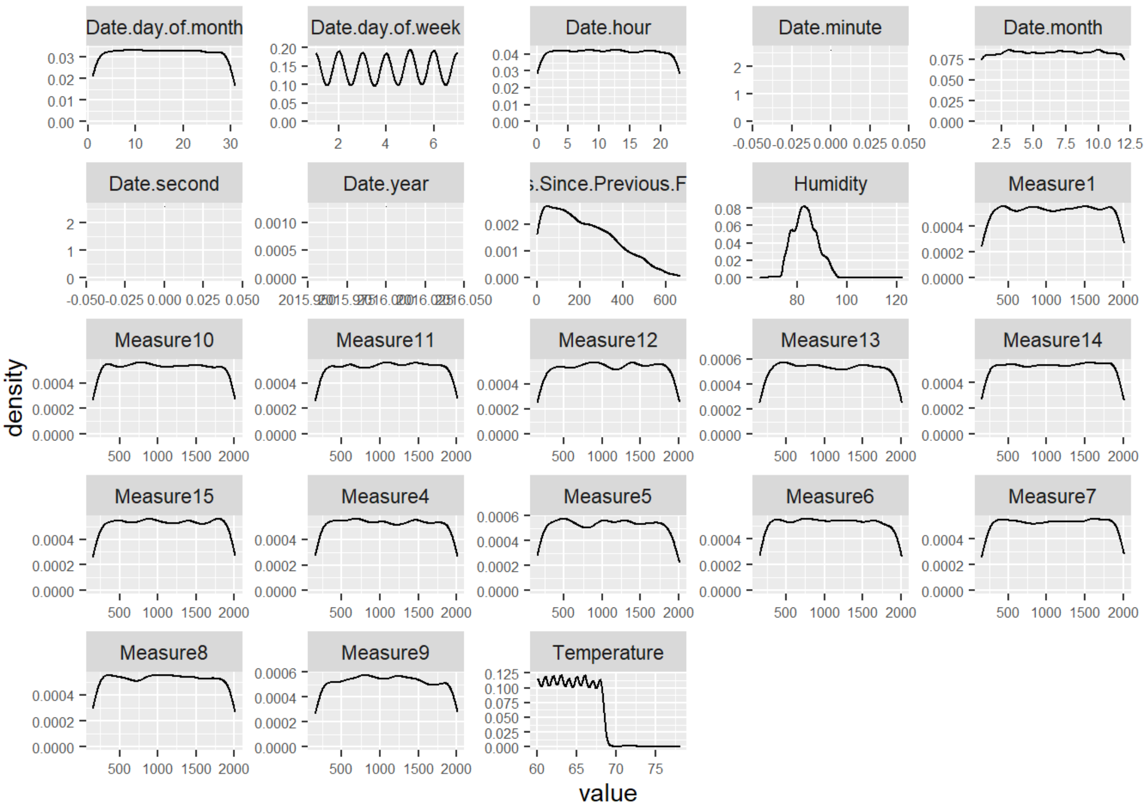



Obtenemos cuatro gráficos ya que son las variables tipo factor que tiene el conjunto de datos. El gráfico Failure está muy desequilibrado ya que el número de fallos es muy bajo respecto al número de no fallos. Después realizaremos operaciones para compensar este desequilibrio.

En los gráficos de Measure 2 y 3 vemos un reparto uniforme de las categorías que nos han salido. En el gráfico de operador, vemos que el número dos tiene mas interacciones que el resto, que son bastante similares.

4.2.3.2 - Analizamos las que son de tipo entero

```
df %>%
  select_if(is.integer) %>%
  gather() %>%
  ggplot(aes(value)) + geom_density() + facet_wrap(~key,scales='free') +
theme(axis.text=element_text(size=6))
```




El gráfico de Horas desde el último fallo nos indica que en las primeras 50 horas de trabajo, las máquinas no suelen fallar, pero cuando vamos aumentando las horas de trabajo, va disminuyendo el número de horas que pasa entre un fallo y el siguiente. El resto de variables presentan patrones coherentes con el sentido de negocio.

Vemos que las variables de fecha, día de la semana, año, mes, minuto y segundo, no nos ofrecen información relevante, así que las eliminamos de nuestro conjunto de datos.

```
df <- df %>%
  select(- c(ï..Date,Date.year,Date.month,Date.day.of.month,Date.day.of.week,Date.hour,Date.minut
e,Date.second))
glimpse(df)
```

```
## Rows: 7,026
## Columns: 20
## $ Temperature                <int> 68, 64, 63, 65, 67, 65, 63, 61, 62, 62, 6~
## $ Humidity                   <int> 77, 76, 80, 81, 76, 80, 80, 83, 81, 76, 8~
## $ Operator                   <fct> Operator1, Operator1, Operator1, Operator~
## $ Measure1                   <int> 1180, 1406, 550, 1928, 1021, 1731, 415, 5~
## $ Measure2                   <fct> 1, 1, 1, 1, 2, 2, 0, 2, 3, 0, 1, 3, 0, 3,~
## $ Measure3                   <fct> 1, 1, 1, 2, 1, 0, 0, 2, 1, 0, 1, 0, 2, 2,~
## $ Measure4                   <int> 1915, 511, 1754, 1326, 185, 1424, 1008, 6~
## $ Measure5                   <int> 1194, 1577, 1834, 1082, 170, 1176, 1086, ~
## $ Measure6                   <int> 637, 1121, 1413, 233, 952, 1223, 1759, 17~
## $ Measure7                   <int> 1093, 1948, 1151, 1441, 1183, 621, 1946, ~
## $ Measure8                   <int> 524, 1882, 945, 1736, 1329, 647, 1814, 64~
## $ Measure9                   <int> 919, 1301, 1312, 1033, 427, 369, 1754, 31~
## $ Measure10                  <int> 245, 273, 1494, 1549, 1638, 239, 1442, 93~
## $ Measure11                  <int> 403, 1927, 1755, 802, 850, 1196, 341, 189~
## $ Measure12                  <int> 723, 1123, 1434, 1819, 379, 1944, 1097, 1~
## $ Measure13                  <int> 1446, 717, 502, 1616, 1529, 1583, 1819, 1~
## $ Measure14                  <int> 719, 1518, 1336, 1507, 755, 1630, 472, 15~
## $ Measure15                  <int> 748, 1689, 711, 507, 844, 237, 491, 1102,~
## $ Hours.Since.Previous.Failure <int> 91, 92, 93, 94, 97, 98, 99, 100, 101, 102~
## $ Failure                    <fct> No, No, No, No, No, No, No, No, No, No, N~
```

Ahora el data frame tiene 7023 filas de 20 variables.

- 4.2.4 - Calidad de datos: Análisis de correlación

Estudiamos la correlación, porque nos interesa que las variables no estén muy correlacionadas para que la modelización posterior sea mas efectiva.

```
df %>%
  select_if(is.integer) %>%
  cor() %>%
  round(digits = 2)
```

```
##                              Temperature Humidity Measure1 Measure4 Measure5
## Temperature                         1.00    -0.05     0.01    -0.02     0.01
## Humidity                           -0.05     1.00     0.00     0.01    -0.03
## Measure1                            0.01     0.00     1.00     0.01     0.03
## Measure4                           -0.02     0.01     0.01     1.00     0.00
## Measure5                            0.01    -0.03     0.03     0.00     1.00
## Measure6                            0.00    -0.01     0.01     0.02     0.00
## Measure7                           -0.01    -0.02     0.00     0.00    -0.01
## Measure8                            0.00     0.02     0.00     0.00    -0.01
## Measure9                           -0.02     0.00    -0.01     0.01    -0.01
## Measure10                          -0.02     0.00     0.01    -0.02    -0.01
## Measure11                           0.01     0.03     0.00     0.00     0.01
## Measure12                           0.00    -0.01     0.00     0.02     0.01
## Measure13                          -0.01    -0.01    -0.02    -0.01     0.00
## Measure14                           0.00     0.00     0.01    -0.01     0.01
## Measure15                          -0.01    -0.01     0.00     0.00     0.01
## Hours.Since.Previous.Failure       -0.01     0.00     0.00    -0.01     0.01
##                              Measure6 Measure7 Measure8 Measure9 Measure10
## Temperature                      0.00    -0.01     0.00    -0.02     -0.02
## Humidity                        -0.01    -0.02     0.02     0.00      0.00
## Measure1                         0.01     0.00     0.00    -0.01      0.01
## Measure4                         0.02     0.00     0.00     0.01     -0.02
## Measure5                         0.00    -0.01    -0.01    -0.01     -0.01
## Measure6                         1.00     0.01     0.00     0.00      0.01
## Measure7                         0.01     1.00     0.01     0.00     -0.02
## Measure8                         0.00     0.01     1.00     0.00     -0.01
## Measure9                         0.00     0.00     0.00     1.00     -0.01
## Measure10                        0.01    -0.02    -0.01    -0.01      1.00
## Measure11                       -0.01     0.01    -0.01     0.00      0.01
## Measure12                        0.02    -0.01    -0.01     0.01      0.01
## Measure13                       -0.01     0.00     0.00    -0.01      0.02
## Measure14                       -0.01     0.01    -0.02    -0.01      0.00
## Measure15                        0.00    -0.01     0.01     0.03     -0.02
## Hours.Since.Previous.Failure    -0.01     0.00     0.00    -0.01     -0.01
##                              Measure11 Measure12 Measure13 Measure14 Measure15
## Temperature                       0.01      0.00     -0.01      0.00     -0.01
## Humidity                          0.03     -0.01     -0.01      0.00     -0.01
## Measure1                          0.00      0.00     -0.02      0.01      0.00
## Measure4                          0.00      0.02     -0.01     -0.01      0.00
## Measure5                          0.01      0.01      0.00      0.01      0.01
## Measure6                         -0.01      0.02     -0.01     -0.01      0.00
## Measure7                          0.01     -0.01      0.00      0.01     -0.01
## Measure8                         -0.01     -0.01      0.00     -0.02      0.01
## Measure9                          0.00      0.01     -0.01     -0.01      0.03
## Measure10                         0.01      0.01      0.02      0.00     -0.02
## Measure11                         1.00     -0.01     -0.01      0.00      0.01
## Measure12                        -0.01      1.00      0.01      0.00      0.01
## Measure13                        -0.01      0.01      1.00      0.01      0.01
## Measure14                         0.00      0.00      0.01      1.00      0.02
## Measure15                         0.01      0.01      0.01      0.02      1.00
## Hours.Since.Previous.Failure      0.00     -0.02      0.00      0.00     -0.01
##                              Hours.Since.Previous.Failure
## Temperature                                         -0.01
## Humidity                                             0.00
## Measure1                                             0.00
## Measure4                                            -0.01
## Measure5                                             0.01
## Measure6                                            -0.01
## Measure7                                             0.00
## Measure8                                             0.00
## Measure9                                            -0.01
## Measure10                                           -0.01
## Measure11                                            0.00
```

```
## Measure12                                     -0.02
## Measure13                                      0.00
## Measure14                                      0.00
## Measure15                                     -0.01
## Hours.Since.Previous.Failure                   1.00
```

Vemos que los valores obtenidos son próximos a cero lo cual indica que las variables no están muy correlaccionadas entre si.

- 4.2.5 - Calidad de datos: Desbalanceo

Vamos a comprobar como está de desbalanceada la variable ‘Failure’.

```
table(df$Failure)
```

```
##
##   No  Yes
## 6961   65
```

Efectivamente vemos que la variable’Failure’, que vamos a utilizar como target está muy desbalanceada y tenemos que corregirlo.

# 4.3 - Trasformación de datos

- 4.3.1 - Corrección del desbalanceo

Para corregir el desbalanceo vamos a utilizar la técnica del inframuestreo, con la que vamos a comprobar la penetración exacta de la target.

```
65/nrow(df) * 100
```

```
## [1] 0.9251352
```

Tenemos 65 ‘Yes’ que sobre el total de los casos supone un 0.92%

Para obtener casi un 10% de ‘Yes’, necesitarímos incrementar la proporción aproximadamente unas 10 veces. Lo que vamos a hacer es reducir los ‘No’ para obtener esa proporción del 10 % de ‘Yes’.

Generamos un nuevo dataframe con los ‘No’ y otro con los ‘Yes’. El de los ‘No’ lo reducimos en tamaño a un 8%.

```
df_nos <- df %>%
  filter(Failure == 'No') %>%
  sample_frac(size = 0.08)
dim(df_nos)
```

```
## [1] 557  20
```

```
df_sis <- df %>% filter(Failure == 'Yes')
dim(df_sis)
```

```
## [1] 65 20
```

Generamos de nuevo un df reducido con los dos nuevos creados. Y comprobamos:

```
df_red <- rbind(df_nos,df_sis)
count(df_red,Failure)
```

```
##   Failure   n
## 1      No 557
## 2     Yes  65
```

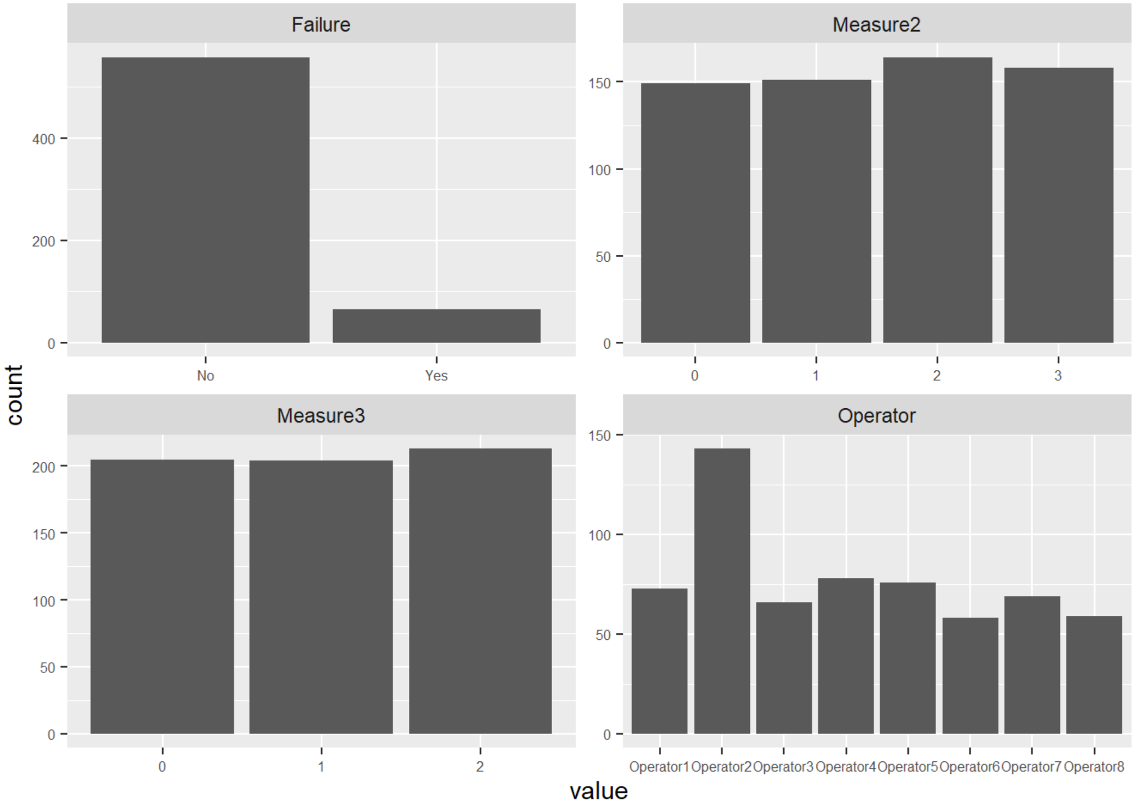
```
65/nrow(df_red) * 100
```

```
## [1] 10.45016
```

De esta manera, hemos obtenido un nuevo dataframe, en el que el 10 % de los datos son 'Yes', eso supone que la variable target tiene un 10% de penetración.

Visualizamos de nuevo los gráficos de las variables tipo factor, para comprobar como hemos aumentado la proporción de 'Yes', respecto al número de 'No'.

```
df_red %>%
  select_if(is.factor) %>%
  gather() %>%
  ggplot(aes(value)) + geom_bar() + facet_wrap(~key,scales='free') + theme(axis.text=element_text
(size=6))
```




# 4.4 - Modelización

- 4.4.1 - Preparacion de las funciones que vamos a necesitar

Función para crear una matriz de confusión.

```
confusion<-function(real,scoring,umbral){
  conf<-table(real,scoring>=umbral)
  if(ncol(conf)==2) return(conf) else return(NULL)
}
```

Funcion para calcular las métricas de los modelos: acierto, precisión, cobertura y F1.

```
metricas<-function(matriz_conf){
  acierto <- (matriz_conf[1,1] + matriz_conf[2,2]) / sum(matriz_conf) *100
  precision <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[1,2]) *100
  cobertura <- matriz_conf[2,2] / (matriz_conf[2,2] + matriz_conf[2,1]) *100
  F1 <- 2*precision*cobertura/(precision+cobertura)
  salida<-c(acierto,precision,cobertura,F1)
  return(salida)
}
```

Función para probar distintos umbrales y ver el efecto sobre precisión y cobertura.

```
umbrales<-function(real,scoring){
  umbrales<-data.frame(umbral=rep(0,times=19),acierto=rep(0,times=19),precision=rep(0,times=19),c
obertura=rep(0,times=19),F1=rep(0,times=19))
  cont <- 1
  for (cada in seq(0.05,0.95,by = 0.05)){
    datos<-metricas(confusion(real,scoring,cada))
    registro<-c(cada,datos)
    umbrales[cont,]<-registro
    cont <- cont + 1
  }
  return(umbrales)
}
```

Funciones que calculan la curva ROC y el AUC.

```
roc<-function(prediction){
  r<-performance(prediction,'tpr','fpr')
  plot(r, col='darkgreen')
}

auc<-function(prediction){
  a<-performance(prediction,'auc')
  return(a@y.values[[1]])
}
```

- 4.4.2 - Creamos las particiones de training (70%) y test (30%)

Generamos una variable aleatoria con una distribución 70-30

```
df_red$random<-sample(0:1,size = nrow(df_red),replace = T,prob = c(0.3,0.7))
```

Creamos los dos dataframes. Y eliminamos la random generada.

```
train<-filter(df_red,random==1)
test<-filter(df_red,random==0)

df_red$random <- NULL
```

- 4.4.3 - Elección del modelo

Vamos a realizar la modelización con Regresión Logística y posteriormente evaluar el modelo.

4.4.3.1 - Identificación de las variables

Concretamos que la variable Target va a ser la variable ‘Failure’.

```
Target <- 'Failure'
```

Eliminamos la variable Target original. Ya que las variables predictoras para el modelo son todas las demas excepto ‘Failure’. Creamos la formula que vamos a pasar al modelo.

```
indep <- names(df_red)[-20]
formula <- reformulate(indep,Target)
```

Vamos a modelizar con una regresión logística.

```
formula_rl <- formula
rl <- glm(formula_rl,train,family=binomial(link='logit'))
summary(rl)
```

```
##
## Call:
## glm(formula = formula_rl, family = binomial(link = "logit"),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6550  -0.2162  -0.1016  -0.0356   3.8763
##
## Coefficients:
##                              Estimate   Std. Error z value   Pr(>|z|)
## (Intercept)              -20.49896304  10.51049358  -1.950    0.05114 .
## Temperature                0.56640882   0.12034471   4.707 0.00000252 ***
## Humidity                  -0.25103259   0.05979821  -4.198 0.00002693 ***
## OperatorOperator2         -1.39100901   0.91974990  -1.512    0.13044
## OperatorOperator3         -0.99775506   1.16687674  -0.855    0.39252
## OperatorOperator4         -1.52834195   1.07862074  -1.417    0.15650
## OperatorOperator5         -1.36317511   1.10131273  -1.238    0.21580
## OperatorOperator6         -1.19273446   1.23251853  -0.968    0.33318
## OperatorOperator7         -0.42138952   0.92687873  -0.455    0.64937
## OperatorOperator8          0.37671257   1.07030425   0.352    0.72486
## Measure1                   0.00067856   0.00055140   1.231    0.21847
## Measure21                 -0.39854697   0.90579237  -0.440    0.65994
## Measure22                 -0.04662488   0.75155147  -0.062    0.95053
## Measure23                 -0.56192845   0.80879446  -0.695    0.48720
## Measure31                  0.16195286   0.66581997   0.243    0.80782
## Measure32                 -0.15233549   0.73261912  -0.208    0.83528
## Measure4                   0.00033710   0.00054980   0.613    0.53979
## Measure5                   0.00006522   0.00050273   0.130    0.89678
## Measure6                   0.00044518   0.00053721   0.829    0.40728
## Measure7                   0.00019586   0.00056964   0.344    0.73098
## Measure8                   0.00062315   0.00058068   1.073    0.28321
## Measure9                  -0.00056341   0.00053740  -1.048    0.29445
## Measure10                  0.00090225   0.00054686   1.650    0.09897 .
## Measure11                 -0.00117944   0.00059055  -1.997    0.04580 *
## Measure12                  0.00033941   0.00054088   0.628    0.53031
## Measure13                 -0.00002838   0.00057555  -0.049    0.96067
## Measure14                  0.00031525   0.00055006   0.573    0.56656
## Measure15                  0.00005054   0.00050998   0.099    0.92106
## Hours.Since.Previous.Failure  -0.00467654   0.00177036  -2.642    0.00825 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 295.20  on 441  degrees of freedom
## Residual deviance: 114.96  on 413  degrees of freedom
## AIC: 172.96
##
## Number of Fisher Scoring iterations: 8
```

Como variables predictivas, con al menos el 90%, sólo resultan 'Temperature', 'Humidity', 'Hours.Since.Previous.Failure', así que las seleccionamos como finales.

```
indep_fin <- c('Temperature','Humidity','Hours.Since.Previous.Failure')
formula_rl <- reformulate(indep_fin,Target)
```

Y volvemos a modelizar

```
rl <- glm(formula_rl,train,family=binomial(link='logit'))
summary(rl)
```

```
##
## Call:
## glm(formula = formula_rl, family = binomial(link = "logit"),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4363  -0.2566  -0.1423  -0.0675   3.2250
##
## Coefficients:
##                                Estimate Std. Error z value   Pr(>|z|)
## (Intercept)                  -12.931454   8.314596  -1.555     0.1199
## Temperature                    0.458075   0.094406   4.852 0.00000122 ***
## Humidity                      -0.239031   0.051025  -4.685 0.00000281 ***
## Hours.Since.Previous.Failure  -0.003380   0.001536  -2.200     0.0278 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 295.20  on 441  degrees of freedom
## Residual deviance: 132.67  on 438  degrees of freedom
## AIC: 140.67
##
## Number of Fisher Scoring iterations: 7
```

Y calculamos el pseudo R cuadrado:

```
pr2_rl <- 1 -(rl$deviance / rl$null.deviance)
pr2_rl
```

```
## [1] 0.5505969
```

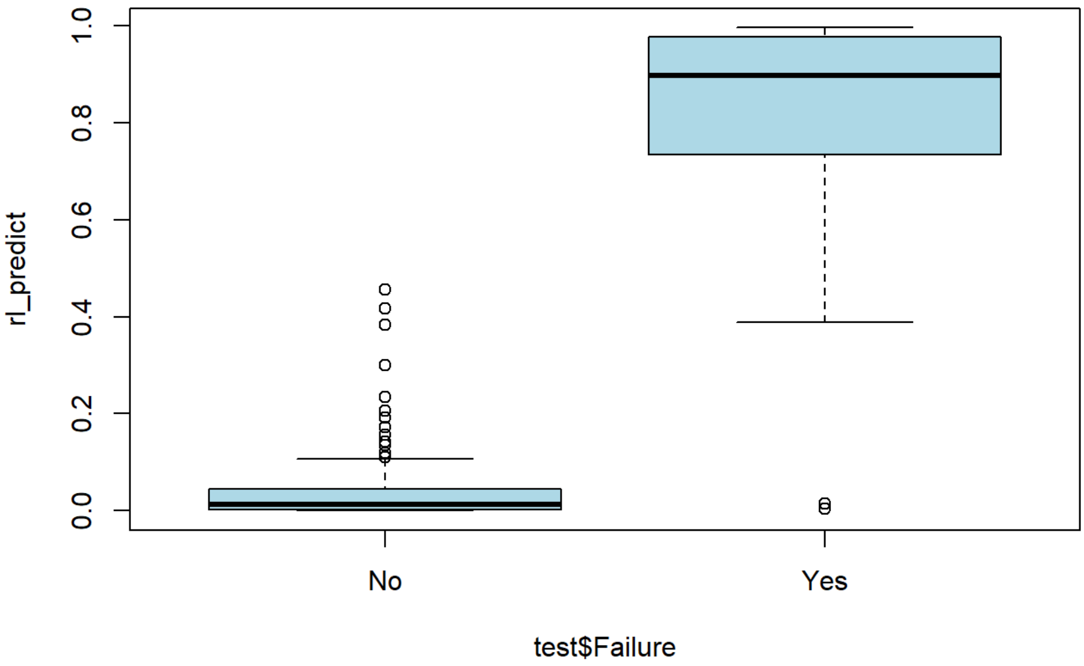
Aplicamos el modelo al conjunto de test, generando un vector con las probabilidades

```
rl_predict<-predict(rl,test,type = 'response')
head(rl_predict)
```

```
##           1           2           3           4           5           6
## 0.005324697 0.013965411 0.003684693 0.045821919 0.016260136 0.001299126
```

Comprobamos en el gráfico:

```
plot(rl_predict~test$Failure, col='lightblue')
```




El modelo está funcionando bien, ya que la probabilidad de que falle la máquina cuando el modelo nos indica que no falla es muy bajo, casi 0, en cambio la probabilidad de que la maqúina falle cuando el modelo indica que falla es muy alto, cerca del 85 %.

Ahora tenemos que transformar la probabilidad en una decisión de si la máquina va a fallar o no.

Con la función umbrales probamos diferentes cortes:

```
umb_rl<-umbrales(test$Failure,rl_predict)
umb_rl
```

```
##    umbral  acierto precision cobertura       F1
## 1    0.05 78.88889  32.07547  89.47368 47.22222
## 2    0.10 90.55556  53.12500  89.47368 66.66667
## 3    0.15 93.88889  65.38462  89.47368 75.55556
## 4    0.20 95.55556  73.91304  89.47368 80.95238
## 5    0.25 96.66667  80.95238  89.47368 85.00000
## 6    0.30 96.66667  80.95238  89.47368 85.00000
## 7    0.35 97.22222  85.00000  89.47368 87.17949
## 8    0.40 97.22222  88.88889  84.21053 86.48649
## 9    0.45 97.77778  94.11765  84.21053 88.88889
## 10   0.50 98.33333 100.00000  84.21053 91.42857
## 11   0.55 98.33333 100.00000  84.21053 91.42857
## 12   0.60 98.33333 100.00000  84.21053 91.42857
## 13   0.65 98.33333 100.00000  84.21053 91.42857
## 14   0.70 97.77778 100.00000  78.94737 88.23529
## 15   0.75 96.66667 100.00000  68.42105 81.25000
## 16   0.80 96.66667 100.00000  68.42105 81.25000
## 17   0.85 96.66667 100.00000  68.42105 81.25000
## 18   0.90 94.44444 100.00000  47.36842 64.28571
## 19   0.95 93.33333 100.00000  36.84211 53.84615
```

Seleccionamos el umbral que maximiza la F1

```
umbral_final_rl<-umb_rl[which.max(umb_rl$F1),1]
umbral_final_rl
```

```
## [1] 0.5
```

Evaluamos la matriz de confusión y las métricas con el umbral optimizado

```
confusion(test$Failure,rl_predict,umbral_final_rl)
```

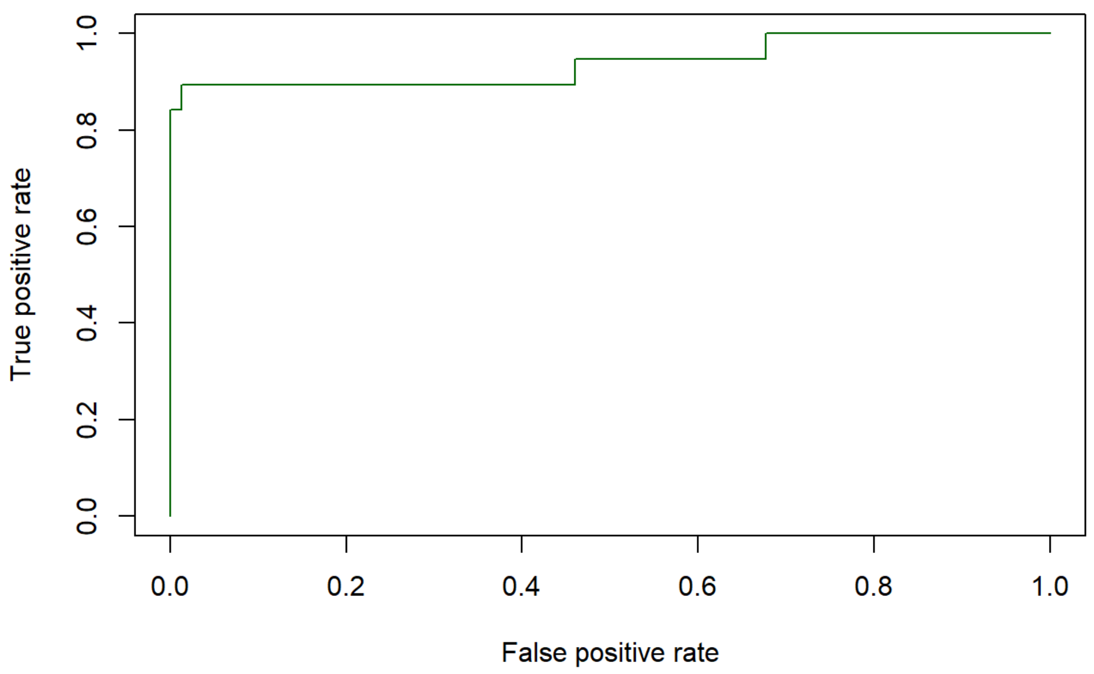
```
##
## real   FALSE TRUE
##   No     161    0
##   Yes      3   16
```

```
rl_metricas<-filter(umb_rl,umbral==umbral_final_rl)
rl_metricas
```

```
##   umbral  acierto precision cobertura       F1
## 1    0.5 98.33333       100  84.21053 91.42857
```

Evaluamos la ROC

```
rl_prediction<-prediction(rl_predict,test$Failure)
roc(rl_prediction)
```




Sacamos las métricas definitivas incluyendo el AUC

```
rl_metricas<-cbind(rl_metricas,AUC=round(auc(rl_prediction),2)*100)
print(t(rl_metricas))
```

```
##                [,1]
## umbral      0.50000
## acierto    98.33333
## precision 100.00000
## cobertura  84.21053
## F1         91.42857
## AUC        94.00000
```

Vemos que nos da un valor de AUC por encima del 90 %, lo que nos dice que es un modelo funciona bien.

Aplicamos el modelo al conjutno de datos completo.

```
df$scoring <- predict(rl,df,type='response')
```

Vamos tomar una decisión de si hay que establecer un mantenimiento preventivo por parte de un operario, y elegimos que nos indique cuando va a ocurrir fallo para un scoring superior al 80 %.

Si el scoring es superior al 80 %, pensamos que la máquina va a fallar.

```
df$prediccion <- ifelse(df$scoring > 0.8,1,0)
table(df$prediccion)
```

```
##
##    0    1
## 6988   38
```

Vamos a contrastar la predicción contra la realidad

```
table(df$prediccion,df$Failure)
```

```
##
##       No  Yes
##   0 6961   27
##   1    0   38
```

Vemos que el número de casos en los que el modelo dice que falla pero no se produce este fallo es cero, por lo tanto podemos bajar el umbral y permitir que el modelo aumente sus fallos sin suponer esto un coste elevado para la empresa.

```
df$prediccion <- ifelse(df$scoring > 0.6,1,0)
```

Vamos a contrastar la predicción contra la realidad

```
table(df$prediccion,df$Failure)
```

```
##
##       No  Yes
##   0 6958   16
##   1    3   49
```

Vemos que el número de veces que el modelo dice que falla pero realmente no lo hace se ha incrementado levemente, siendo aún sostenible, e incluso al número de veces que el modelo dice que no falla y finalmente si falla también disminuye, lo que produce una disminución de casos en los que se envíe una persona de mantenimiento y no falle finalmente la máquina.

En esta tabla vemos los casos en los que modelo nos predice que la máquina va a fallar y las condicones del resto de las variables para el momento de ese fallo.

```
df[grep('1',df$prediccion),]
```

```
##      Temperature Humidity  Operator Measure1 Measure2 Measure3 Measure4
## 415           72       67 Operator3      935        3        0     1228
## 416           70       68 Operator3     1479        0        1      738
## 438           71       77 Operator6     1559        2        1     1990
## 1296          71       65 Operator4     1025        1        1      342
## 1298          69       70 Operator6      437        3        0     1303
## 1470          68       71 Operator2      655        2        1     1542
## 1471          69       72 Operator2     1550        1        1     1292
## 1472          69       69 Operator2      199        1        2      885
## 1473          68       69 Operator2      837        0        1      794
## 1474          67       71 Operator2      993        3        2      538
## 1475          67       72 Operator2     1008        2        0     1405
## 1476          67       70 Operator4      455        3        0      244
## 1961          71       69 Operator7     1323        2        2     1339
## 1962          72       72 Operator7      742        2        2     1773
## 1963          71       73 Operator7      550        2        1     1417
## 2430          68       71 Operator8      792        1        2      395
## 2544          72       72 Operator1      831        2        0      920
## 2545          68       71 Operator1      706        1        1     1012
## 2546          68       67 Operator1      441        2        1     1715
## 2547          69       69 Operator1     1975        1        1     1702
## 2548          73       72 Operator1      999        0        1      891
## 2549          71       73 Operator3      595        2        2     1645
## 3017          72       71 Operator2     1370        1        0     1457
## 3018          71       73 Operator2      970        0        0      219
## 3671          72       75 Operator7     1789        1        1     1167
## 3672          69       71 Operator2      920        2        1      946
## 3673          74       69 Operator2      876        0        0      273
## 3674          71       72 Operator2     1268        2        0      466
## 3675          73       74 Operator2     1865        0        1     1226
## 4096          77       77 Operator1     1558        0        0     1586
## 4739          73       69 Operator2      725        3        1      708
## 4740          75       68 Operator4     1222        0        2      430
## 5140          72       72 Operator4     1289        3        1      461
## 5141          76       77 Operator4     1687        2        2     1190
## 5142          72       69 Operator4      588        1        2     1639
## 5143          68       74 Operator4     1421        3        1      393
## 5144          68       72 Operator6     1137        3        2      321
## 5145          68       73 Operator6      802        1        1     1959
## 5146          69       75 Operator6      454        1        0     1621
## 5384          72       72 Operator7      323        2        1      244
## 5385          73       77 Operator7     1079        3        0      780
## 5386          71       74 Operator7      666        2        1     1338
## 5387          70       69 Operator7      379        2        0      413
## 5388          69       75 Operator7     1859        0        2     1095
## 6009          78       77 Operator5      881        3        1      333
## 6010          75       73 Operator5      319        0        0     1861
## 6011          77       70 Operator5     1753        3        2     1478
## 6012          72       70 Operator2     1727        0        2      493
## 6013          69       68 Operator2     1009        2        0     1359
## 6014          71       75 Operator2     1795        3        2      251
## 6541          76       75 Operator6      732        2        1      605
## 7016          77       69 Operator2      869        0        1     1492
##      Measure5 Measure6 Measure7 Measure8 Measure9 Measure10 Measure11 Measure12
## 415      1226     1883      831      853      417      1189       497       697
## 416      1686      447     1446     1053     1977      1985      1571      2005
## 438       570     1475      999     1871      968      1740       531      1667
## 1296     1154      912     1400     1814      525       194       493       247
## 1298     1926      970     1533      512      551      1412       595       882
## 1470     1869      826      566     1413     1057      1598      1100      1318
## 1471      167     1432      606     1531     1177      1782       386       815
## 1472     1651     1935     1475      970     1051      1706      1604       968
## 1473     1788     1432     1542     1128      779      1476       554       509
```

```
## 1474     994     1873     772     818     1877     1850      744      1839
## 1475    1617      969     827     419      335     1616      852      1960
## 1476     565     1819    1263    1414     1429      271     1445       912
## 1961     971      507    1098    1117     1224      170     1033      1107
## 1962     766     1226    1177    1473      785     1798     1232       561
## 1963     269      724    1873     511      915     1097      796      1692
## 2430    1838     2000    1007    1370     1664      329      651       952
## 2544    1956     1993    1999     220     1926      246      293       602
## 2545     676     1304     723     705      457      742     1060      1233
## 2546    1805     1688     742     871     1650     1640      295      1767
## 2547     365      260    1265     254      824      581      432      1546
## 2548     608      829     695     398     1019     1090     1487       411
## 2549     837     1910    1085     172     1190     1742     1110      1684
## 3017    1943     1094     579     291     2001     1982      904      1723
## 3018    1214     1002    1891    1524     1141      770     1655       972
## 3671    1834      565    1794    1055     1475      921     1505       578
## 3672    1546      215    1096     322      160     1598     1633      1100
## 3673    1471     1990     334    1971      682     1507      646       950
## 3674    1291      841    1154     295     1277      377      464      1360
## 3675     695     1943    1821    1324     1716     1528      454      1613
## 4096     251     1869    1987     452     1401     1040     1216      1392
## 4739     220      403     913     686     1882     1015      885      1188
## 4740     246      184    1988    1508     1712      241     1453      1222
## 5140    1343      822     173    1146      992     1243      465      1376
## 5141    1398     1697    1338    1187     1275      549     1106       894
## 5142     617     1190    1605     602     1815      661     1774      1383
## 5143    1876      435    2007     382     1011      698      577      1045
## 5144     799     1656    1397    1174      194     1532     1732      1465
## 5145     711     1461     486    1619     1510     1969      733       911
## 5146     773      360     375     844      924     1709      469       642
## 5384    1272      405    1346    1100      220      817     1722       174
## 5385     568     1492     691     191      587     1115      642       926
## 5386    1311     1730    1819     443      370     1625     1151      1634
## 5387    1984     1950     498     166      514     1441     1756      1677
## 5388    1349     1077    1597    1197      535     1323     1599       241
## 6009    1024     1317    1887     752     1185     1985     1562      1842
## 6010     656      716     904    1157     1100      309      637       537
## 6011    1072     1021     378     493      301      380     1193      1269
## 6012    1470      586     633    1442     1273     1774      236       614
## 6013     309     1267     891    1257      498     1540      769       424
## 6014    1613     1365     813    1341      949     1947     1284      1881
## 6541    1198     1033     915     938      387     1255     1151       611
## 7016    1779      410    1447     535     1774      564     1799      1766
##      Measure13 Measure14 Measure15 Hours.Since.Previous.Failure Failure
## 415        576      1082       452                            1     Yes
## 416       1305       457      1566                            1     Yes
## 438       1605      1260      1558                           28      No
## 1296      1595      1906      1510                          510     Yes
## 1298      1348      1103       283                            2     Yes
## 1470       993      1984       529                          194      No
## 1471      1873       666      1693                          195     Yes
## 1472      1312      1307       795                            1     Yes
## 1473      1382      1311      1169                            1     Yes
## 1474       929      1109      1652                            1     Yes
## 1475      1540       534      1066                            1     Yes
## 1476       594      1093      1054                            1     Yes
## 1961       719       715      1617                          593     Yes
## 1962       914      1911      1279                            1     Yes
## 1963      1117      1906      1159                            2     Yes
## 2430      1324      1139       829                          203      No
## 2544      1645       971       785                          353     Yes
## 2545       995      1459      1402                            2     Yes
## 2546      1710       157       396                            1     Yes
## 2547       608      1867       303                            1     Yes
```

```
## 2548      1562       759       602                             1     Yes
## 2549       416      1750      1902                             1     Yes
## 3017      1226      1888       666                             1     Yes
## 3018       505      1103       244                             1     Yes
## 3671       953      1080       266                             1     Yes
## 3672       607       523      1779                             1     Yes
## 3673       855       517       326                             1     Yes
## 3674      1856      1138      1491                             1     Yes
## 3675      1863       871      1178                             1     Yes
## 4096       258       740       542                           522     Yes
## 4739      1331       462      1741                             1     Yes
## 4740      1738       688      1884                             2     Yes
## 5140      1303      1362      1346                             1     Yes
## 5141       601      1850      1247                             1     Yes
## 5142      1644      1681       508                             1     Yes
## 5143      1789      1845      1762                             1     Yes
## 5144      1815      1630      1383                             1     Yes
## 5145      1618      1364      1545                             1     Yes
## 5146      1678       300      1870                             1     Yes
## 5384       744      1276       573                           296     Yes
## 5385       560      1101       965                             1     Yes
## 5386       483       305      1031                             1     Yes
## 5387       414       658       362                             1     Yes
## 5388      1556      1809       439                             1     Yes
## 6009      1447      1153       449                             1     Yes
## 6010       717       445      1054                             1     Yes
## 6011       795      1153       706                             1     Yes
## 6012      1884       432      1147                             1     Yes
## 6013      1674      1942       528                             1     Yes
## 6014      1154      1066      1231                             1     Yes
## 6541      1893       582       360                           666     Yes
## 7016       986      1627      1885                           178     Yes
##        scoring prediccion
## 415  0.9825622          1
## 416  0.9466645          1
## 438  0.7487351          1
## 1296 0.9114171          1
## 1298 0.8740029          1
## 1470 0.6435367          1
## 1471 0.6913444          1
## 1472 0.8983694          1
## 1473 0.8482783          1
## 1474 0.6867617          1
## 1475 0.6332053          1
## 1476 0.7357616          1
## 1961 0.7492081          1
## 1962 0.9446098          1
## 1963 0.8943422          1
## 2430 0.6365283          1
## 2544 0.8384296          1
## 2545 0.7755151          1
## 2546 0.9001797          1
## 2547 0.8983694          1
## 2548 0.9642378          1
## 2549 0.8946612          1
## 3017 0.9558666          1
## 3018 0.8946612          1
## 3671 0.8927627          1
## 3672 0.8456880          1
## 3673 0.9886779          1
## 3674 0.9151571          1
## 3675 0.9435547          1
## 4096 0.8975765          1
## 4739 0.9822165          1
```

```
## 4740 0.9943100         1
## 5140 0.9446098         1
## 5141 0.9699248         1
## 5142 0.9721713         1
## 5143 0.6285510         1
## 5144 0.7318576         1
## 5145 0.6824463         1
## 5146 0.6780992         1
## 5384 0.8628603         1
## 5385 0.8908342         1
## 5386 0.8699178         1
## 5387 0.9332246         1
## 5388 0.6780992         1
## 6009 0.9877471         1
## 6010 0.9815046         1
## 6011 0.9963334         1
## 6012 0.9649206         1
## 6013 0.9182097         1
## 6014 0.8403992         1
## 6541 0.8460357         1
## 7016 0.9947568         1
```

# 5 Conclusiones

Se ha trabajado sobre un histórico de mediciones de sensores y se ha obtenido un modelo predictivo de alta calidad.

El modelo es estable y consigue plasmar las características de los momentos en los que se producen los fallos de la máquina.

Con los datos objenidos, sabiendo que las variables de Temperatura, Humedad y Horas desde el úlitmo fallo, sabemos que con ciertas condiciones de estos indicadores la máquina fallará.

# 6 Resultados

Nuestra previsión es que usando este modelo se puede construir un plan de mantenimiento preventivo sobre la maquinaria y predecir cada cuantas horas hay que revisarla o cambiar piezas susceptibles de avería.

En conjunto con el departamento de Mantenimiento se realiza el Plan de Mantenimiento Preventivo en el que se incluirán un conjunto de intervenciones u operaciones preventivas de debemos realizar en los equipos, para lograr unos objetivos de disponibilidad, fiabilidad y coste y por lo tanto ampliar la vida útil de los equipos.

Con este plan vamos a lograr:

- Reducir las intervenciones correctivas, ya que con una buena planificación y previsión se evitarán averías.
- Reducir los gastos de reparaciones, tanto materiales como humanos.
- Aumentar la disponibilidad de los activos, por lo que conseguiremos una mayor rentabilidad en la producción.
- Reducción de costes por reemplazo de equipos, puesto que la vida útil de la maquinaria se verá ampliada.
- Aumentar la productividad y reducir costes derivados de la parada de producción.
- Reducir riesgos de accidentes laborales relacionados con fallos en equipos.
- Evitar sanciones por incumplimiento de la normativa de reglamentación de instalaciones.