People Analytics

Pedro Costa Ferreira, Bianca Gonçalves e Matheus Camelo - IBRE/FGV

Fevereiro de 2018

— 1. Dados para insight de decisão —

1.1. Compreensão do negócio

Nosso exemplo diz respeito a uma grande empresa que quer entender porque alguns de seus melhores e mais experientes funcionários estão saindo prematuramente. A empresa também deseja prever quais bons funcionários pretendem deixar a companhia futuramente. Os dados foram extraídos da Kaggle.[1]

1.2. Solução analítica

Temos dois objetivos principais: primeiro, queremos entender porque os bons funcionários estão saindo e, em segundo lugar, queremos prever quem serão os próximos a deixarem a empresa.

Portanto, a proposta é trabalhar com o departamento de RH para reunir dados relevantes sobre os funcionários e para comunicar o efeito significativo que poderia explicar e prever a saída dos funcionários.

1.3. Leitura dos dados e pacotes

```
# Importando os dados
hr <- read.csv2("HR_comma_sep.csv", sep = ",", stringsAsFactors = F)
```

Infelizmente, os gerentes não mantiveram um registro organizado com os motivos que levaram as pessoas a deixarem seus empregos, mas ainda é possível encontrar algumas explicações no conjunto de dados fornecido pelo departamento de RH.

Para os 15000 funcionários, tem-se conhecimento das seguintes variáveis: nível de satisfação dos funcionários (*satisfaction_level*), última avaliação (*last_evaluation*) (anual), número de projetos (*number_project*), horas mensais médias (*average_montly_hours*), tempo de empresa (*time_spend_company*) (em anos), se os funcionários tiveram um acidente de trabalho nos últimos 2 anos (*Work_accident*), se o funcionário deixou a empresa (*left*), se os funcionários tiveram uma promoção nos últimos 5 anos (*promotion_last_5years*), departamento (*sales*) e salário (*salary*).

```
names(hr)
```

```
##  [1] "satisfaction_level"    "last_evaluation"
##  [3] "number_project"        "average_montly_hours"
##  [5] "time_spend_company"    "Work_accident"
##  [7] "left"                  "promotion_last_5years"
##  [9] "sales"                 "salary"
```

[1] https://www.kaggle.com/ludobenistant/hr-analytics-1/notebook

Alguns pacotes serão utilizados ao longo do estudo sendo necessária a instalação dos mesmos:

- install.packages("dplyr"): **A Grammar of Data Manipulation** (Wickham, Francois, Henry, Müller, (2017)).
- install.packages("corrplot"): **Visualization of a Correlation Matrix** (Wei and Simko (2017)).
- install.packages("caret"): **Classification and Regression Training** (Kuhn (2017)).
- install.packages("rpart"): **Recursive Partitioning and Regression Trees** (Therneau, Atkinson, Ripley (2018)).
- install.packages("caTools"): **Tools: moving window statistics, GIF, Base64, ROC AUC, etc** (Tuszynski (2014))

O carregamentos dos pacotes, utilizando a função *library()*, será feito ao longo do texto para que fique claro onde estamos utlizando cada biblioteca.

1.4. Tabela de base analítica

As variáveis nível de satisfação (*satisfaction_level*) e última avaliação (*last_evaluation*) estão na base classificadas como "character". Assim, foi necessário mudar a classe dessas variáveis para o tipo mais adequado ("numeric").

```
# Verificando a classe das variáveis
# class(hr$satisfaction_level)
# class(hr$last_evaluation)

# Mudando a classe das variáveis
hr$satisfaction_level <- as.numeric(hr$satisfaction_level)
hr$last_evaluation <- as.numeric(hr$last_evaluation)
```

A seguir, um recorte da base de dados que será utilizada no estudo:

```
head(hr)
```

```
##   satisfaction_level last_evaluation number_project average_montly_hours
## 1               0.38            0.53              2                  157
## 2               0.80            0.86              5                  262
## 3               0.11            0.88              7                  272
## 4               0.72            0.87              5                  223
## 5               0.37            0.52              2                  159
## 6               0.41            0.50              2                  153
##   time_spend_company Work_accident left promotion_last_5years sales salary
## 1                  3             0    1                     0 sales    low
## 2                  6             0    1                     0 sales medium
## 3                  4             0    1                     0 sales medium
## 4                  5             0    1                     0 sales    low
## 5                  3             0    1                     0 sales    low
## 6                  3             0    1                     0 sales    low
```

— 2. Análise exploratória dos dados —

Nesta fase, queremos entender os dados que serão utilizados, avaliar analiticamente e visualizar graficamente as variáveis que compõem a base.

2.1. Relatório da qualidade dos dados

A tabela a seguir contém uma breve análise descritiva dos dados. Podemos ver diferentes medidas estatísticas de tendência central, como por exemplo, a média dos funcionários que saíram (*left*) é aproximadamente igual a 24%, o nível de satisfação (*satisfaction_level*) é de cerca de 62% e a média de desempenho (*last_evaluation*) é de cerca de 71%. Verificamos que, em média, as pessoas trabalham em 3 a 4 projetos (*number_project*) por ano e cerca de 200 horas por mês (*average_montly_hours*).

```
summary(hr)
```

```
##  satisfaction_level last_evaluation  number_project  average_montly_hours
##  Min.   :0.0900     Min.   :0.3600   Min.   :2.000   Min.   : 96.0
##  1st Qu.:0.4400     1st Qu.:0.5600   1st Qu.:3.000   1st Qu.:156.0
##  Median :0.6400     Median :0.7200   Median :4.000   Median :200.0
##  Mean   :0.6128     Mean   :0.7161   Mean   :3.803   Mean   :201.1
##  3rd Qu.:0.8200     3rd Qu.:0.8700   3rd Qu.:5.000   3rd Qu.:245.0
##  Max.   :1.0000     Max.   :1.0000   Max.   :7.000   Max.   :310.0
##  time_spend_company Work_accident         left
##  Min.   : 2.000     Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 3.000     1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 3.000     Median :0.0000   Median :0.0000
##  Mean   : 3.498     Mean   :0.1446   Mean   :0.2381
##  3rd Qu.: 4.000     3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :10.000     Max.   :1.0000   Max.   :1.0000
##  promotion_last_5years    sales              salary
##  Min.   :0.00000       Length:14999       Length:14999
##  1st Qu.:0.00000       Class :character   Class :character
##  Median :0.00000       Mode  :character   Mode  :character
##  Mean   :0.02127
##  3rd Qu.:0.00000
##  Max.   :1.00000
```

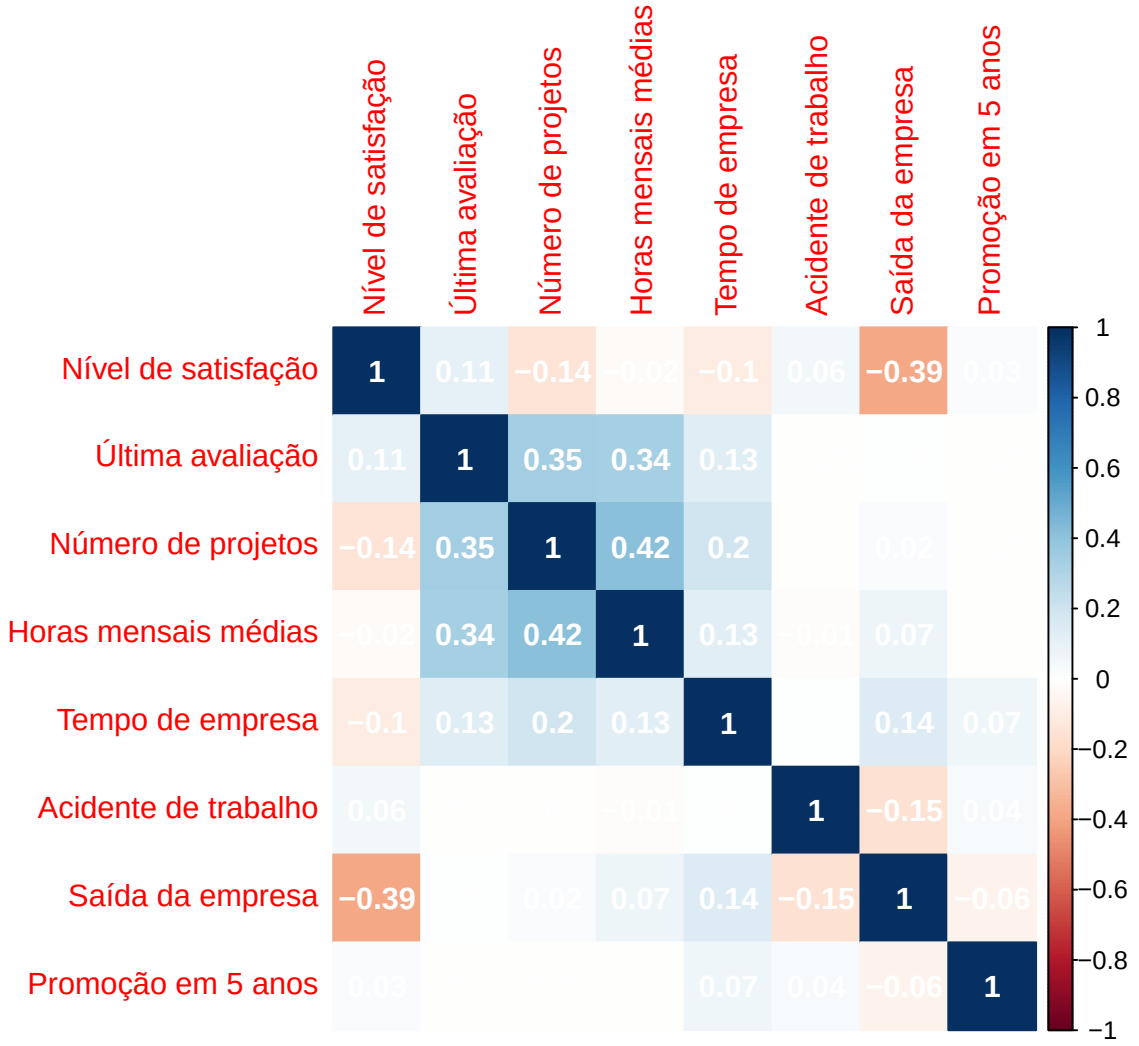
2.2. Primeiras visualizações

2.2.1. Gráfico

O gráfico a seguir apresenta as correlações entre cada variável da base de dados. A cor representa a direção da relação (positiva - em azul, ou negativa - em vermelho).

```
library(dplyr)
HR_correlation <- hr %>% select(satisfaction_level:promotion_last_5years)
# as variáveis "sales" e "salary" não foram utilizadas pois são variáveis qualitativas
names(HR_correlation) <- c("Nível de satisfação", "Última avaliação",
                           "Número de projetos", "Horas mensais médias",
                           "Tempo de empresa", "Acidente de trabalho",
                           "Saída da empresa", "Promoção em 5 anos")
```

```
library(corrplot)
M <- cor(HR_correlation)
corrplot(M, method = "color", tl.cex = 1, type = "full", addCoef.col = "white")
```



É possível observar que as variáveis não são altamente correlacionadas tanto positivamente quanto negativamente. A maior correlação positiva (0,42) é entre o número de projetos em que o funcionário já trabalhou e os horas mensais médias, ou seja, quanto mais projetos, mais horas de trabalho.

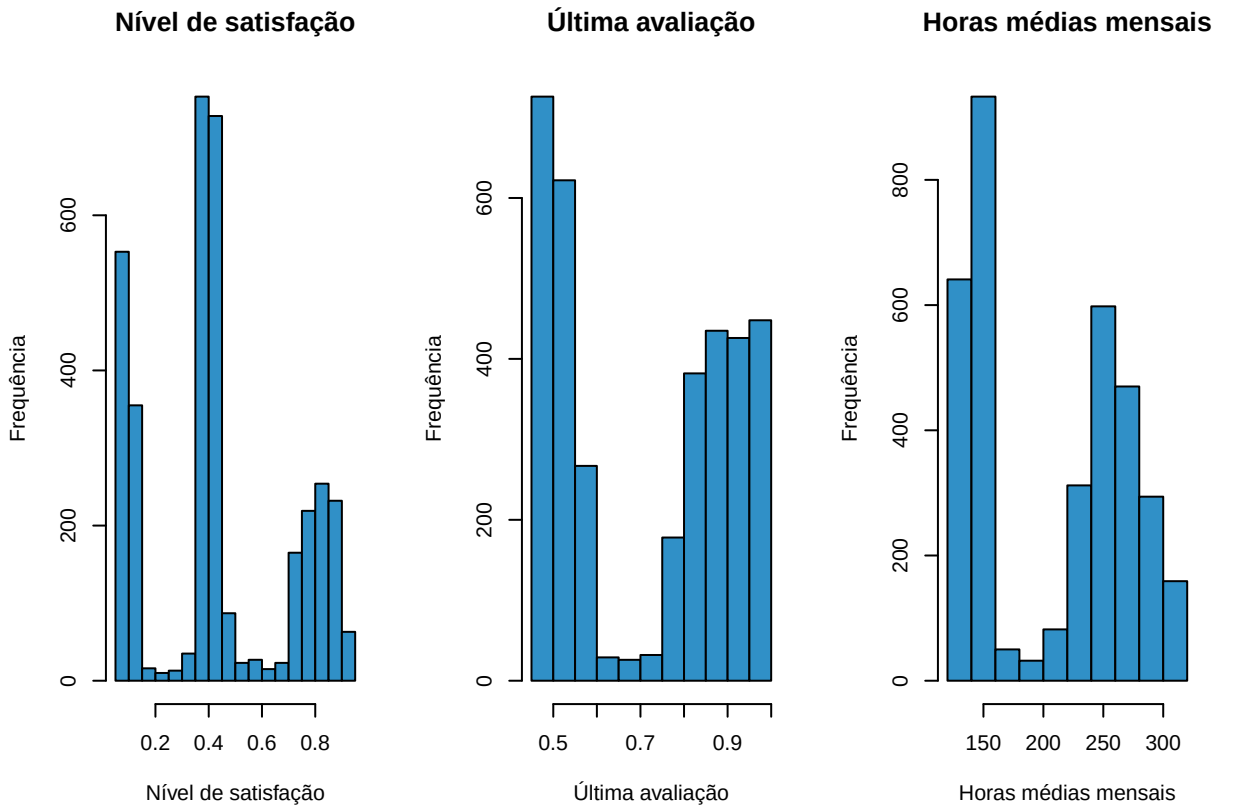
A maior correlação negativa (-0,39) é entre o nível de satisfação do funcionário em relação à empresa e o fato do mesmo ter deixado de trabalhar. Em outras palavras, quanto menor for o nível de satisfação do funcionário, maior a chance dele sair da empresa.

2.3. Quem está deixando a empresa?

Neste momento, levaremos em consideração apenas os **funcionários que deixaram a empresa** com o objetivo de visualizar melhor a distribuição de determinadas variáveis da nossa base de dados.

Podemos observar nos histogramas a seguir que não são todos os funcionários que a empresa deve manter. Algumas pessoas não são bem avaliadas a partir da variável última avaliação (*last_evaluation*), mas claramente há também muitos bons colaboradores que saem. Além disso, notamos que a variável nível de satisfação (*satisfaction_level*) segue uma distribuição trimodal, no entanto, não nos diz muita coisa pois representa que há bons e maus colaboradores na empresa. Já na terceira variável, horas mensais médias (*average_montly_hours*), o histograma apresenta dois extremos predominantes: tanto funcionários que trabalham menos quanto os que trabalham mais estão deixando a empresa.

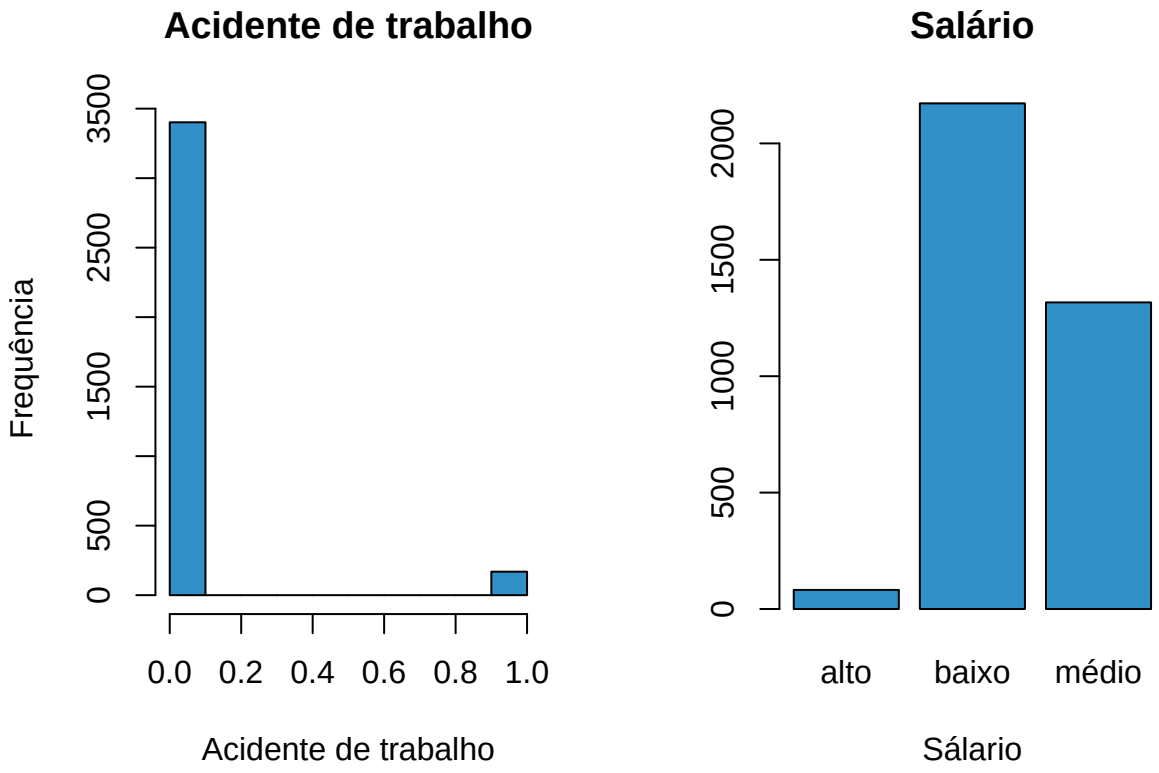
```
hr_left <- hr %>% filter(left==1) # apenas funcionários que saíram
par(mfrow=c(1,3))
hist(hr_left$satisfaction_level, col= "#3090C7", main = "Nível de satisfação",
     xlab = "Nível de satisfação", ylab = "Frequência")
hist(hr_left$last_evaluation, col= "#3090C7", main = "Última avaliação",
     xlab = "Última avaliação", ylab = "Frequência")
hist(hr_left$average_montly_hours, col= "#3090C7", main = "Horas médias mensais",
     xlab = "Horas médias mensais",  ylab = "Frequência")
```



Continuando nossa análise, verificamos também que pouquíssimas pessoas que deixaram a empresa tiveram acidente de trabalho e àquelas com os menores salários posssuem uma tendência maior a sair. Graficamente temos:

```r
par(mfrow=c(1,2))
hist(hr_left$Work_accident, col = "#3090C7", main = "Acidente de trabalho",
     xlab = "Acidente de trabalho", ylab = "Frequência")

barplot(table(hr_left$salary), col = "#3090C7", main = "Salário",
        names.arg = c("alto", "baixo","médio"), xlab = "Sálario")
```



Por fim, do total de 15000 funcionários que compõem o banco de dados, 3571 deixaram a empresa:

```r
nrow(hr_left)
```

```
## [1] 3571
```

A seguir, é apresentado o total de funcionários que receberam uma **avaliação acima da média**, ou **passaram pelo menos quatro anos na empresa**, ou **trabalhavam em mais de 5 projetos** ao mesmo tempo e ainda sim deixaram a empresa. Para nós, essas são as pessoas que a empresa deveria ter mantido.

```r
hr_good_leaving_people <- hr_left %>% filter(last_evaluation >= 0.70 |
                                                time_spend_company >= 4 |
                                                number_project > 5)
nrow(hr_good_leaving_people)
```

```
## [1] 2014
```

2.4. Por que bons colaboradores estão saindo da empresa?

Vamos criar uma base de dados que contenha apenas os melhores funcionários e tentar entender suas principais características.

```
hr_good_people <- hr %>% filter(last_evaluation >= 0.70 | time_spend_company >= 4
                                        | number_project > 5)
summary(hr_good_people)
```

```
##  satisfaction_level last_evaluation  number_project  average_montly_hours
##  Min.   :0.090      Min.   :0.3600   Min.   :2.000   Min.   : 96.0
##  1st Qu.:0.490      1st Qu.:0.7300   1st Qu.:3.000   1st Qu.:171.0
##  Median :0.680      Median :0.8300   Median :4.000   Median :218.0
##  Mean   :0.617      Mean   :0.8015   Mean   :4.159   Mean   :211.8
##  3rd Qu.:0.830      3rd Qu.:0.9100   3rd Qu.:5.000   3rd Qu.:253.0
##  Max.   :1.000      Max.   :1.0000   Max.   :7.000   Max.   :310.0
##  time_spend_company Work_accident         left
##  Min.   : 2.000     Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 3.000     1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 4.000     Median :0.0000   Median :0.0000
##  Mean   : 3.916     Mean   :0.1521   Mean   :0.2061
##  3rd Qu.: 5.000     3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :10.000     Max.   :1.0000   Max.   :1.0000
##  promotion_last_5years    sales              salary
##  Min.   :0.00000       Length:9772        Length:9772
##  1st Qu.:0.00000       Class :character   Class :character
##  Median :0.00000       Mode  :character   Mode  :character
##  Mean   :0.02384
##  3rd Qu.:0.00000
##  Max.   :1.00000
```

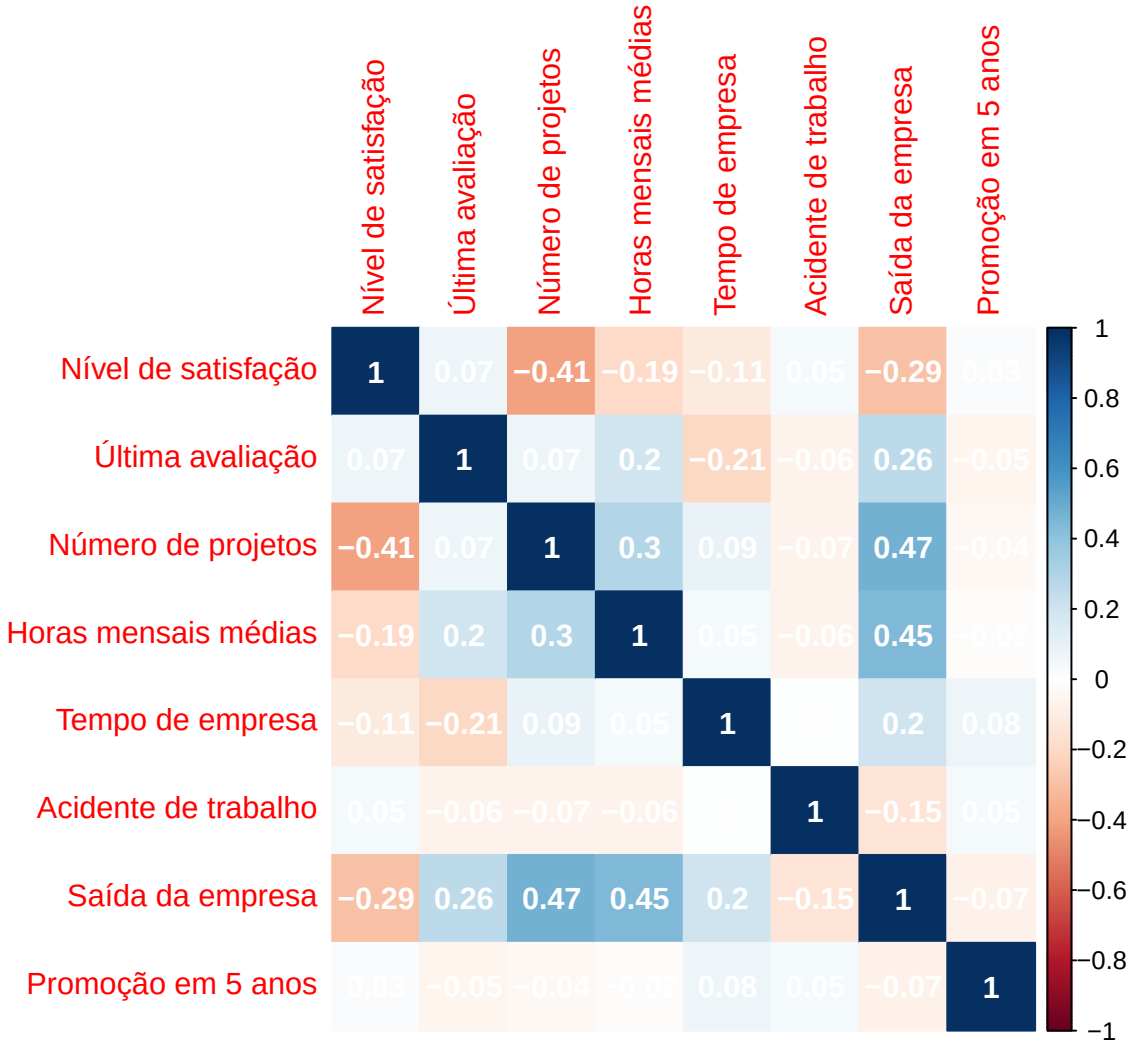
Comparando ao primeiro resumo dos dados que considera a base completa, verificamos que a média de projetos (*number_project*) e de horas trabalhadas (*average_montly_hours*) aumentou ao levar em conta os melhores funcionários, de 3,803 para 4,159 e de 201,1 para 211,8 respectivamente. O mesmo acontece com a variável última avaliação (*last_evaluation*), em que a média aumentou de 0,7161 para 0,8015. Resultado plausível uma vez que agora estamos considerando os melhores funcionários.

Outro resultado importante é na variável de saída da empresa (*left*), onde o percentual de saída dos funcionários passou de aproximadamente 24% para 20%, mostrando que 1 em cada 5 bons funcionários deixaram a empresa. O gráfico de correlações a seguir intensifica a análise nessa e nas demais variáveis comentadas.

Gráfico de correlações

```
hr_good_people_select <- hr_good_people %>%
  select(satisfaction_level: promotion_last_5years)

names(hr_good_people_select) <- c("Nível de satisfação", "Última avaliação",
                                  "Número de projetos", "Horas mensais médias",
                                  "Tempo de empresa", "Acidente de trabalho",
                                  "Saída da empresa", "Promoção em 5 anos")
M <- cor(hr_good_people_select)
corrplot(M, method="color", tl.cex = 1, type = "full", addCoef.col = "white")
```



Pela variável Saída da empresa (*left*), no gráfico de correlações, fica muito mais claro que, em média, os bons funcionários que saem não são satisfeitos, trabalham em muitos projetos, passam muitas horas na empresa a cada mês e não são promovidos.

— 3. Modelagem —

Agora, queremos prever quais dos bons funcionários serão os próximos a deixarem a empresa.

3.1. Selecionar o banco de dados

Vamos usar o mesmo banco de dados da seção anterior, onde foi mantido os melhores funcionários de acordo com as variáveis última avaliação (*last_evaluation*), tempo de empresa (*time_spend_company*) e número de projetos (*number_project*).

```
hr_model <- hr %>% filter(last_evaluation >= 0.70 | time_spend_company >= 4 |
                            number_project > 5)
# summary(hr_model)
```

3.2. Modelagem preditiva

Depois de configurar a validação cruzada, iremos construir e comparar diferentes modelos preditivos. O primeiro usa um modelo de árvore (*Tree learning*), o segundo *Naives Bayes* e o terceiro uma regressão logística (*Logistic regression*).

Para definir o melhor modelo dentre os três citados, iremos avaliar a estatística *Kappa* e a precisão de cada um deles, pois essas são as métricas padrão, suportadas pelo pacote caret, utilizadas para avaliar o desempenho de algoritmos de *machine learning*.

A estatística *Kappa* compara uma precisão observada com uma precisão esperada (chance aleatória). Nesse estudo, os modelos foram construídos utilizando *machine learning* e, em cada um deles, é possível observar sua matriz de confusão. Desta forma, as colunas de cada matriz representam um "avaliador" (os valores reais de cada instância a ser classificada) e as linhas, a outro "avaliador" (o classificador de *machine learning* utilizado para realizar a classificação).

Sabemos que, quanto mais próxima de 1 se encontra estatística *Kappa*, melhor é o modelo. Porém as classificações atribuídas aos intervalos da mesma variam de acordo com a literatura, assim como as possíveis interpretações. Não existe uma interpretação padronizada para essa estatística. Sabe-se apenas que pelo menos duas considerações devem ser levadas em consideração no momento em que a mesma for interpretada: a primeira é que uma matriz de confusão deve sempre ser usada para comparação com a estatística *Kappa* buscando obter uma interpretação mais precisa; a segunda é que os valores da estatística podem variar no contexto em que estão sendo avaliados.

Matriz de confusão

Uma matriz de confusão é um resumo dos resultados obtidos a partir de um modelo preditivo em um problema de classificação. Em outras palavras, ela apresenta as formas pelas quais o modelo de classificação testado foi confundido na hora de realizar as previsões. Com isso, é possível notar os tipos de erros que estão sendo realizados e não só os erros gerados pelo seu classificador.

É possível calcular uma matriz de confusão a partir dos seguintes passos:

1. É necessário um conjunto de dados teste ou um de validação com resultados esperados;
2. Uma previsão deve ser feita para cada linha do conjunto de dados teste;
3. Contar, a partir dos resultados esperados e das previsões:
    a) O número de previsões corretas para cada classe;
    b) O número de predições incorretas para cada classe, organizadas pela classe que foram previstas.

Então, esses números são organizados em uma matriz (ou tabela) em que suas linhas correspondem a uma classe esperada e suas colunas, a uma classe real (de referência).

Figure 1: Matriz de confusão

Somando os valores das células que são diagonalmente adjacentes, é possível determinar a exatidão geral do modelo. A diagonal principal mostra o número de previsões corretas, e a secundária mostra o número de previsões incorretas. Desta forma, a matriz que apresentar a menor soma (FP+FN) representa o melhor ajuste, uma vez que o modelo "errou" menos nos dados de previsão (Figura 1).

Validação cruzada

Muitas técnicas em um projeto de *machine learning* fazem com que nos deparemos com *overfitting* e, para evitar esse problema, é aplicado o processo de validação. Existem diversos tipos de validação e, neste estudo, será utilizado o que divide os dados em três bases diferentes: base de treino (k partições), validação e teste.

A validação cruzada é um tipo de validação que vem sendo muito utilizada nos últimos tempos. Ela estima o erro do método de aprendizado em observações que não são utilizadas na base de treino previamente definida (ou seja, os dados pertencentes à base de validação), em outras palavras, ela é capaz de estimar o comportamento do modelo em relação a novos dados (desde que a probabilidade conjunta das variáveis independentes e dependente seja a mesma da utilizada durante o treino).

O desempenho é calculado como:

- Taxa de acerto (%): número de exemplos classificados corretamente/total de exemplos;
- Taxa de erro (%): número de exemplos classificados incorretamente/total de exemplos;

Com base nesses valores o algoritmo é treinado n vezes com parâmetros diferentes. Esses n algoritmos são avaliados em relação à taxa de acerto e então a média dessa taxa é calculada, e logo depois a variância também. Uma das vantagens desse método é a redução da variabilidade ao se usar k partições, diferentemente dos demais métodos que normalmente dividem em apenas dois grupos. Para utilizá-lo é necessário que a variável de interesse, saída da empresa (*left*), esteja classificada como fator:

```
# Defina a variável de interesse como um fator
hr_model$left <- as.factor(hr_model$left)
```

A seguir, determina-se o método de reamostragem, o número de repartições e o número de repetições no processo de validação cruzada:

```r
# install.packages("caret")
library(caret)
train_control <- trainControl(method = "cv", number = 5, repeats = 3)
head(train_control)
```

```
## $method
## [1] "cv"
##
## $number
## [1] 5
##
## $repeats
## [1] 3
##
## $search
## [1] "grid"
##
## $p
## [1] 0.75
##
## $initialWindow
## NULL
```

3.2.1 Aprendizagem de árvores

As árvores de decisão são um dos modelos mais utilizados em inferência indutiva. É um instrumento de apoio à tomada de decisão que consiste em uma representação gráfica, a partir de uma decisão inicial, de todas alternativas possíveis. Esse método faz com que problemas vistos como muito complexos sejam repartidos em problemas muito mais simples, e podem ser novamente repartidos, até que eles sejam solucionados de forma simples e eficaz.

A representação gráfica da árvore de decisão geralmente é feita a partir de linhas para identificar a decisão e nós para identificar as questões sobre as quais se deve decidir. Cada ramo (formado por linhas e nós) termina numa "folha" que identifica a consequência mais provável da sequência de decisões tomadas.

Em nossa modelagem, a variável alvo é Saída da empresa (*left*) e as demais variáveis são as variáveis de entrada.

```r
library(rpart)

# Treinar o modelo
rpartmodel <- train(left~., data = hr_model, trControl = train_control, method = "rpart")
# Fazer previsões
predictions <- predict(rpartmodel, hr_model)
hr_model_tree <- cbind(hr_model, predictions)
# Resumir os trabalhos
confusionMatrix <- confusionMatrix(hr_model_tree$predictions, hr_model_tree$left)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction    0    1
##          0 7591  276
##          1  167 1738
##
##                Accuracy : 0.9547
##                  95% CI : (0.9504, 0.9587)
##     No Information Rate : 0.7939
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8586
##  Mcnemar's Test P-Value : 2.878e-07
##
##             Sensitivity : 0.9785
##             Specificity : 0.8630
##          Pos Pred Value : 0.9649
##          Neg Pred Value : 0.9123
##              Prevalence : 0.7939
##          Detection Rate : 0.7768
##    Detection Prevalence : 0.8051
##       Balanced Accuracy : 0.9207
##
##        'Positive' Class : 0
##
```

3.2.2 Naives Bayes

É uma técnica de classificação baseada no teorema de Bayes com uma suposição de independência entre preditores (recursos) (Ray 2017). Não é o único algoritmo para treinar classificadores, mas é uma família de algoritmos baseados em um princípio comum: todos os classificadores Bayes ingênuos assumem que o valor de um recurso particular é independente do valor de qualquer outro recurso, dada a variável de classe (variável de interesse).

O teorema de bayes pode ser definido como:

$$P(c|d) = \frac{P(d|c)P(c)}{P(c)}$$

onde c representa a classe e d os preditores.

O interesse principal no método é verificar as probabilidades da(s) classe(s) $P(c|d)$ no conjunto de treinamento. Como é considerado independência entre os preditores, o treinamento se torna mais rápido pois as probabilidades condicionais $P(d_1|c).P(d_2|c)...P(d_n|c)$ são mais fáceis de serem calculadas.

O método funciona bem quando os preditores são variáveis categóricas em comparação com as variáveis numéricas. Para esse último caso, a distribuição gaussiana é assumida (que é uma forte hipótese). Já em casos mais complexos envolvendo variáveis numéricas, funções de Kernel podem ser utilizadas na modelagem ao invés da distribuição normal.

```r
# treinar o modelo
e1071model2 <- train(left~., data = hr_model, trControl = train_control, method = "nb")
# fazer previsões
```

```r
predictions <- predict(e1071model2, hr_model)
e1071modelbinded <- cbind(hr_model, predictions)
# resumir os resultados
confusionMatrix <- confusionMatrix(e1071modelbinded$predictions, e1071modelbinded$left)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7663  601
##          1   95 1413
##
##                Accuracy : 0.9288
##                  95% CI : (0.9235, 0.9338)
##     No Information Rate : 0.7939
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.76
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9878
##             Specificity : 0.7016
##          Pos Pred Value : 0.9273
##          Neg Pred Value : 0.9370
##              Prevalence : 0.7939
##          Detection Rate : 0.7842
##    Detection Prevalence : 0.8457
##       Balanced Accuracy : 0.8447
##
##        'Positive' Class : 0
##
```

3.2.3 Regressão logística

O modelo de regressão logística é semelhante ao modelo de regressão linear. No entanto, no modelo logístico a variável resposta Y_i é binária. Uma variável binária assume dois valores, como por exemplo, $Y_i = 0$ e $Y_i = 1$, denominados "fracasso" e "sucesso", respectivamente.

Na modelagem a seguir foi utilizada a metodologia de *Boosting*, um conjunto de algoritmos usados para classificação que apresenta um custo computacional relativamente baixo. Funciona através da combinação sequencialmente de classificadores simples, dando maior peso em cada passo às observações classificadas incorretamente no passo anterior do conjunto de dados de treinamento (Rübesam and Dias, n.d.). Após a combinação, um classificador final é produzido.

Os algoritmos ajustam modelos aditivos através da otimização de um critério. Em regressão logística, o critério utilizado é a verossimilhança da binomial. O Algoritmo LogitBoost otimiza a log-verossimilhança da binomial através de um algoritmo do tipo Newton-Rhapson. Ou seja, eles são combinações lineares de classificadores simples, os quais são construídos de maneira a dar mais peso aos erros cometidos.

```r
# install.packages("caTools")
library(caTools)
```

```r
gmlmodel <- train(left~., data=hr_model, trControl = train_control, method = "LogitBoost")
# fazer previsões
predictions <- predict(gmlmodel, hr_model)
gmlmodelbinded <- cbind(hr_model, predictions)
# resumir os resultados
confusionMatrix <- confusionMatrix(gmlmodelbinded$predictions, gmlmodelbinded$left)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7676  335
##          1   82 1679
##
##                Accuracy : 0.9573
##                  95% CI : (0.9531, 0.9612)
##     No Information Rate : 0.7939
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8632
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9894
##             Specificity : 0.8337
##          Pos Pred Value : 0.9582
##          Neg Pred Value : 0.9534
##              Prevalence : 0.7939
##          Detection Rate : 0.7855
##    Detection Prevalence : 0.8198
##       Balanced Accuracy : 0.9115
##
##        'Positive' Class : 0
##
```

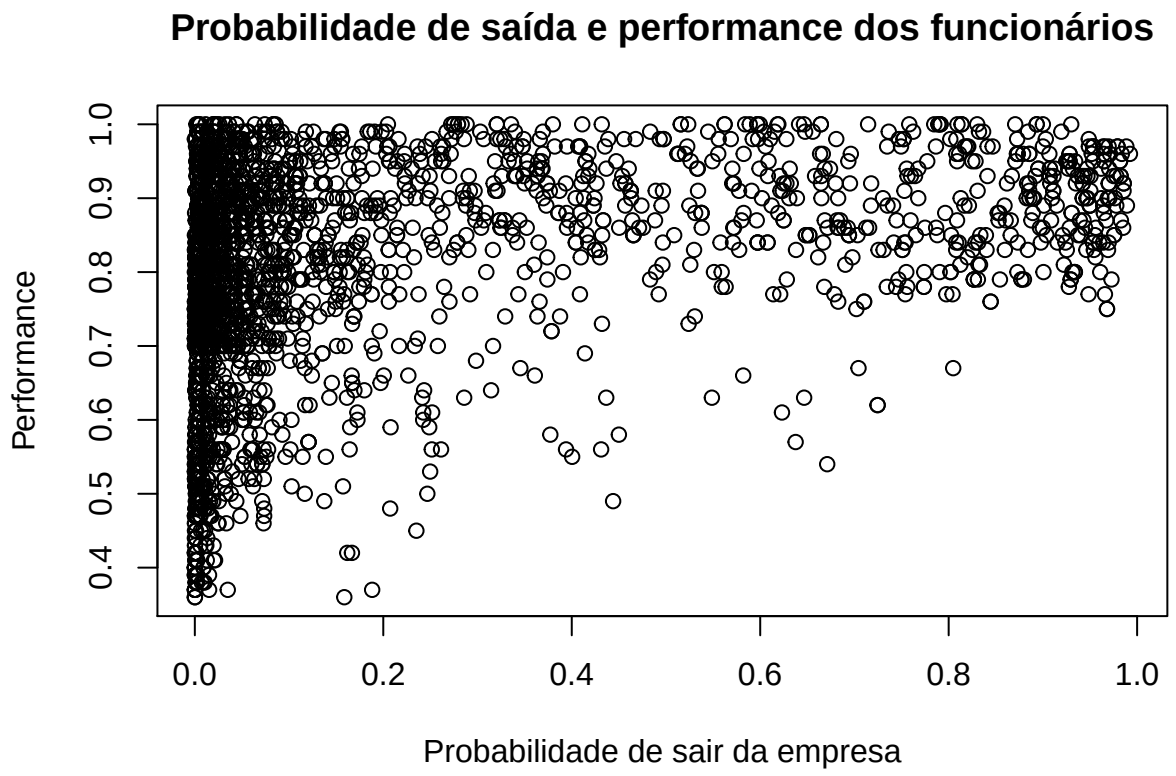
— 4. Insights úteis —

A matriz de confusão e a precisão dos diferentes modelos mostram que o poder preditivo é muito parecido e parece robusto. Apesar disso, verificamos que a matriz de confusão do modelo logístico apresentou melhores resultados, ou seja, obteve uma menor soma dos elementos que compõem a diagonal secundária da mesma. Ademais, apresentou o melhor Kappa (0,8674). Assim, decidiu-se manter o modelo de regressão logística pela simplicidade do mesmo e por ter apresentado os melhores resultados.

A seguir um gráfico que mostra a probabilidade de saída dos funcionários e seu desempenho. Precisamos nos concentrar no canto superior direito. Para fazer isso, foi construída uma tabela de dados classificados pela probabilidade de deixar a empresa encontrada no modelo de regressão logística e também pelo desempenho, definindo para a empresa qual deveria ser seu "funcionário prioridade" na hora de manter no cargo.

```r
set.seed(100)
#  Mantenha alguns dados para testar novamente o modelo final
```

```
inTraining <- createDataPartition(hr_model$left, p = .75, list = FALSE)
training <- hr_model[ inTraining,]
testing  <- hr_model[-inTraining,]
# Estimar os fatores de desgaste
logreg = glm(left ~ ., family = binomial(logit), data = training)
# Faça previsões sobre os dados fora da amostra
probaToLeave = predict(logreg, newdata = testing, type = "response")
# Estrutura a saída de previsão em uma tabela
predattrition = data.frame(probaToLeave)
# Adicione uma coluna ao dataframe de tempo de desgaste que contém o desempenho
predattrition$performance = testing$last_evaluation
plot(predattrition$probaToLeave, predattrition$performance,
     xlab = "Probabilidade de sair da empresa", ylab = "Performance",
     main = "Probabilidade de saída e performance dos funcionários")
```



A tabela a seguir apresenta os primeiros 10 funcionários que a empresa deveria ter mantido. Note que para classificar os colaboradores, foi criada uma coluna chamada *Prioridade* definida como o produto da probabilidade de sair e a perfomance do funcionário. Uma sugestão seria enviar esse resultado para os diferentes gerentes por departamento e avisá-los quais bons funcionários podem sair em breve para assim a empresa tentar mantê-los e consequentemente diminuir o turnover.

```
# Criando a variável prioridade
predattrition$priority <- predattrition$probaToLeave * predattrition$performance

# Ordenando o data.frame em ordem decrescente
```

```r
predattrition2 <- predattrition[order(predattrition$priority, decreasing=TRUE),]

head(predattrition2, 10)
```

```
##      probaToLeave performance  priority
## 9707    0.9885638        0.97 0.9589069
## 767     0.9923916        0.96 0.9526959
## 586     0.9793298        0.97 0.9499499
## 458     0.9790388        0.97 0.9496677
## 822     0.9837824        0.96 0.9444311
## 8677    0.9697847        0.97 0.9406912
## 591     0.9657600        0.97 0.9367872
## 9464    0.9639238        0.97 0.9350061
## 8156    0.9818180        0.95 0.9327271
## 9632    0.9818180        0.95 0.9327271
```

Referências

Ray, Sunil. 2017. “6 Easy Steps to Learn Naive Bayes Algorithm.” https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/.

Rübesam, Alexandre, and Ronaldo Dias. n.d. “Alocaçao de Clientes Em Grupos Usando Classificação via Boosting: Uma Comparaçao Com Os Métodos Tradicionais de Classificação.”