APRENDIZADO DE MÁQUINA COM R

Table of contents

Bem-vindo

Este curso tem como objetivo propagar as ideias básicas de aprendizado de máquina e previsão no *software* estatístico R. A ideia principal é cobrir as técnicas mais usadas como regressão linear, árvores de decisão, e também detalhes básicos e aspectos práticos do aprendizado de máquina. Inicialmente será utilizado alguns códigos básicos do R para alguns modelos de previsão. Contudo, o foco principal será no pacote caret, o qual tem a finalidade de tornar as técnicas de aprendizado mais simples, combinando um grande número de preditores que foram construídos no R.

Pré-requisitos

Os pré-requisitos que serão úteis para o curso são: análise exploratória de dados no R, programação básica em R e conhecimentos teóricos básicos sobre modelos de regressão.

Part I

Introdução

O que é o Aprendizado de Máquina?

Em 1959, Arthur Samuel definiu o aprendizado de máquina como o "campo de estudo que dá aos computadores a habilidade de aprender sem serem explicitamente programados". Ou seja, é um método de análise de dados que automatiza a construção de modelos analíticos. É baseado na ideia de que sistemas podem aprender com dados, identificar padrões e tomar decisões com o mínimo de intervenção humana. A importância desse aprendizado se deve principalmente ao fato de que atualmente tem surgido cada vez mais a necessidade de manipulações de grandes volumes e variedades de dados disponíveis.

Para que serve?

Com o aprendizado de máquina é possível produzir, rápida e automaticamente, modelos capazes de analisar dados maiores e mais complexos, e entregar resultados mais rápidos e precisos – mesmo em grande escala.

Onde é usado?

Ao construir modelos precisos há mais chances de identificar boas oportunidades e de evitar riscos desconhecidos. Na prática, podemos citar alguns exemplos reais do uso de aprendizado de máquina:

- Os governos locais podem tentar prever os pagamentos de pensão no futuro para que eles saibam se seus mecanismos de geração de receita têm fundos suficientes gerados para cobrir esses pagamentos de pensão.
- O Google pode querer prever se você vai clicar em um anúncio para que ele possa mostrar apenas os anúncios com maior probabilidade de receber cliques e, assim, aumentar a receita.
- A Amazon, a Netflix e outras empresas como essa mostram um filme e querem que você veja um próximo filme. Para fazer isso, eles querem mostrar a você o que você pode estar interessado, para que eles possam mantê-lo assistindo e, novamente, aumentar a receita.
- As seguradoras empregam grandes grupos de atuários e estatísticos para tentar prever seu risco de todo tipo de coisas diferentes, como por exemplo a morte.

Como funciona?

A funcionalidade do aprendizado de máquina se resume a tentar prever um certo modelo para o conjunto de dados em questão. Há dois modos de isso ser feito: pelo aprendizado supervisionado e pelo aprendizado não supervisionado. Veremos a definição de cada um deles a seguir.

Tipos de Aprendizado de Máquina

Aprendizado não supervisionado

Na aprendizagem não supervisionada, temos um conjunto de dados não rotulados e queremos de alguma forma agrupá-los por um certo padrão encontrado. Vejamos alguns exemplos:

- **Exemplo 1**: Dada uma imagem de homem/mulher, temos de prever sua idade com base em dados da imagem.
- **Exemplo 2**: Dada as informações sobre que músicas uma pessoa costuma ouvir, sugerir outras que possam agradá-la também.

Aprendizado supervisionado

No aprendizado supervisionado, por outro lado, temos um conjunto de dados já rotulados que sabemos qual é a nossa saída correta e que deve ser semelhante ao conjunto. Queremos assim, com base nesses dados, ser capaz de classificar outros dados do mesmo tipo e que ainda não foram rotulados.

- **Exemplo 1**: Dada uma coleção de 1000 pesquisas de uma universidade, encontrar uma maneira de agrupar automaticamente estas pesquisas em grupos que são de alguma forma semelhantes ou relacionadas por diferentes variáveis, tais como a frequência das palavras, frases, contagem de páginas, etc.
- **Exemplo 2**: Dada uma grande amostra de e-mails, encontrar uma maneira de agrupá-los automaticamente em "spam" ou "não spam", de acordo com as características das palavras, tais como a frequência com que uma certa palavra aparece, a frequência de letras maiúsculas, de cifrões ($), entre outros.

Se os valores da variável rótulo, também chamada de variável de interesse, são valores discretos finitos ou ainda categóricos, então temos um problema de classificação e o algoritmo que criaremos para resolver nosso problema será chamado **Classificador**.

Se os valores da Variável de Interesse são valores contínuos, então temos um problema de regressão e o algoritmo que criaremos será chamado **Regressor**.

A aprendizagem supervisionada será o principal foco do curso.

Predição

Queremos então construir um algoritmo ”preditor” capaz de inferir se um dado pertence ou não a uma certa categoria. O preditor será formado dos seguintes componentes:

Pergunta → Amostra de entrada → Características → Algoritmo → Parâmetros → Avaliação

Pergunta

O nosso objetivo é responder a uma pergunta de tipo ”O dado A é do tipo x ou do tipo y?”. Por exemplo, podemos querer saber se é possível detectar automaticamente se um e-mail é um spam ou um ”ham”, isto é, não spam. O que na verdade queremos saber é: ”É possível usar características quantitativas para classificar um e-mail como spam?”.

Amostra de Entrada

Uma vez formulada a pergunta, precisamos obter uma amostra de onde tentaremos extrair informações que caracterizam a categoria a qual um dado pertence e então usar essas informações para classificar outros dados não categorizados. O ideal é que se tenha uma amostra grande, assim teremos melhores parâmetros para construir nosso preditor.

No caso da pergunta sobre um e-mail ser spam ou não, temos acesso a base de dados ”spam” disponível no pacote ”kernlab”, onde cada linha dessa base é um e-mail e nas colunas temos a porcentagem de palavras e números contidos em cada e-mail e, entre outras coisas, a nossa variável de interesse ”type” que classifica o e-mail como spam ou não:

```
library(kernlab)
data(spam)
head(spam)
```

```
  make address  all num3d  our over remove internet order mail receive will
1 0.00    0.64 0.64     0 0.32 0.00   0.00     0.00  0.00 0.00    0.00 0.64
2 0.21    0.28 0.50     0 0.14 0.28   0.21     0.07  0.00 0.94    0.21 0.79
3 0.06    0.00 0.71     0 1.23 0.19   0.19     0.12  0.64 0.25    0.38 0.45
```

```
4 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.31 0.63    0.31 0.31
5 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.31 0.63    0.31 0.31
6 0.00    0.00 0.00     0 1.85 0.00   0.00     1.85  0.00 0.00    0.00 0.00
  people report addresses free business email  you credit your font num000
1   0.00   0.00      0.00 0.32     0.00  1.29 1.93   0.00 0.96    0   0.00
2   0.65   0.21      0.14 0.14     0.07  0.28 3.47   0.00 1.59    0   0.43
3   0.12   0.00      1.75 0.06     0.06  1.03 1.36   0.32 0.51    0   1.16
4   0.31   0.00      0.00 0.31     0.00  0.00 3.18   0.00 0.31    0   0.00
5   0.31   0.00      0.00 0.31     0.00  0.00 3.18   0.00 0.31    0   0.00
6   0.00   0.00      0.00 0.00     0.00  0.00 0.00   0.00 0.00    0   0.00
  money hp hpl george num650 lab labs telnet num857 data num415 num85
1  0.00  0   0      0      0   0    0      0      0    0      0     0
2  0.43  0   0      0      0   0    0      0      0    0      0     0
3  0.06  0   0      0      0   0    0      0      0    0      0     0
4  0.00  0   0      0      0   0    0      0      0    0      0     0
5  0.00  0   0      0      0   0    0      0      0    0      0     0
6  0.00  0   0      0      0   0    0      0      0    0      0     0
  technology num1999 parts pm direct cs meeting original project   re  edu
1          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00
2          0    0.07     0  0   0.00  0       0     0.00       0 0.00 0.00
3          0    0.00     0  0   0.06  0       0     0.12       0 0.06 0.06
4          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00
5          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00
6          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00
  table conference charSemicolon charRoundbracket charSquarebracket
1     0          0          0.00            0.000                 0
2     0          0          0.00            0.132                 0
3     0          0          0.01            0.143                 0
4     0          0          0.00            0.137                 0
5     0          0          0.00            0.135                 0
6     0          0          0.00            0.223                 0
  charExclamation charDollar charHash capitalAve capitalLong capitalTotal type
1           0.778      0.000    0.000      3.756          61          278 spam
2           0.372      0.180    0.048      5.114         101         1028 spam
3           0.276      0.184    0.010      9.821         485         2259 spam
4           0.137      0.000    0.000      3.537          40          191 spam
5           0.135      0.000    0.000      3.537          40          191 spam
6           0.000      0.000    0.000      3.000          15           54 spam
```

Obtida a amostra, precisamos dividi-la em duas partes que chamaremos de *Conjunto de Treino* e *Conjunto de Teste*. O conjunto de treino será usado para construir o algoritmo. É dele que vamos extrair as informações que julgarmos utéis para classificar uma categoria de dado. É importante que o modelo de previsão seja feito com base **apenas** no conjunto de treino.

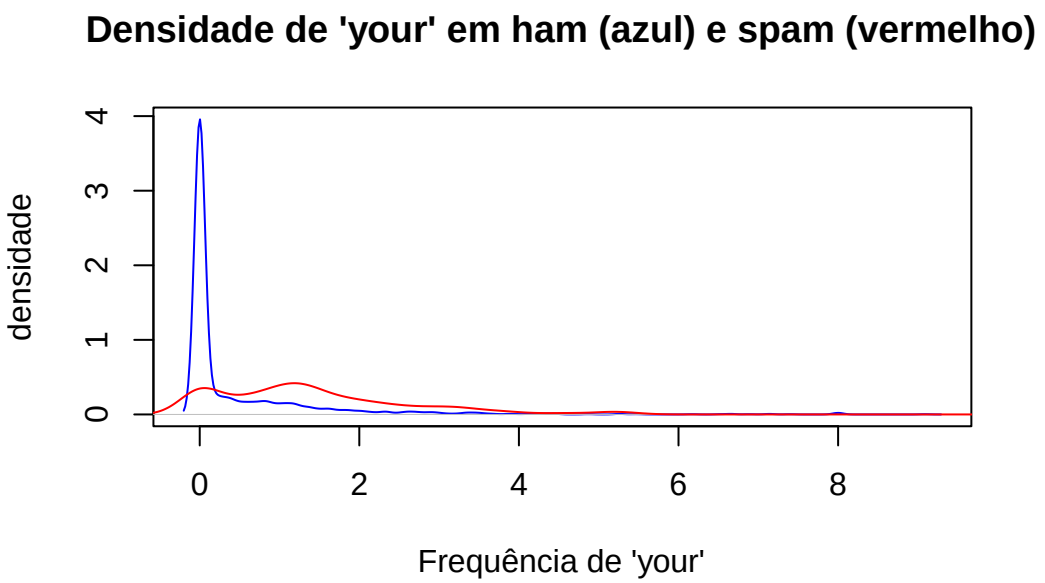
```r
set.seed(127)
indices = sample(dim(spam)[1], size = 2760)
treino = spam[indices,]
teste = spam[-indices,]
```

Após construido o algoritmo, usaremos o conjunto de teste para obter a estimativa de erro, que será detalhada mais a frente.

Características

Temos que encontrar agora características que possam indicar a categoria dos dados. Podemos, por exemplo, vizualizar algumas variáveis graficamente para obter uma ideia do que podemos fazer. No nosso exemplo de e-mails, podemos querer avaliar se a frequência de palavras "your" em um e-mail pode indicar se ele é um spam ou não.

```r
plot(density(treino$your[treino$type=="nonspam"]), col="blue",
     main = "Densidade de 'your' em ham (azul) e spam (vermelho)",
     xlab = "Frequência de 'your'", ylab = "densidade")
lines(density(treino$your[treino$type=="spam"]), col="red")
```



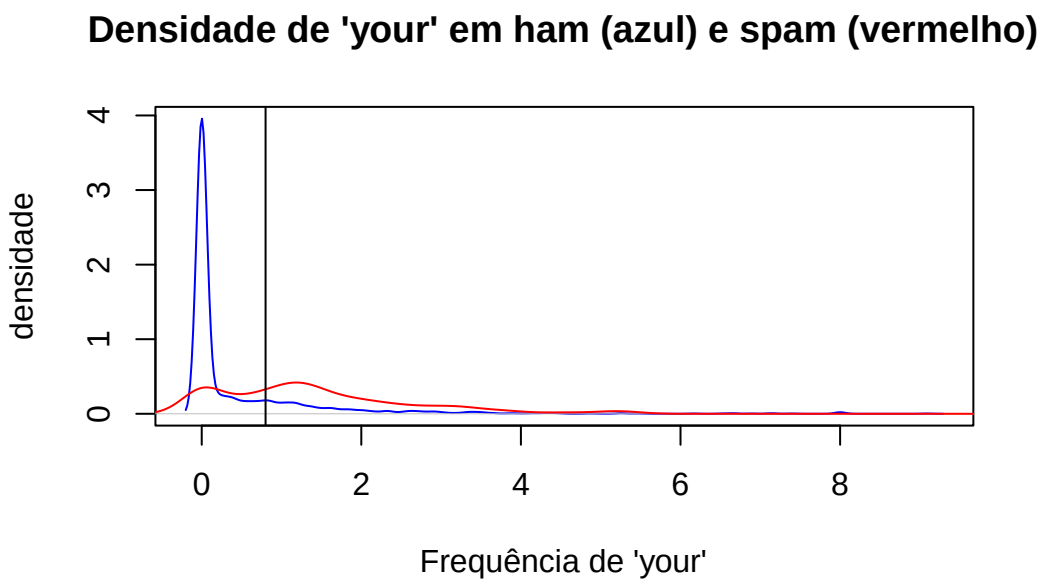
Pelo gráfico podemos notar que a maioria dos e-mails que são spam têm uma frequência maior da palavra ”your”. Por outro lado, aqueles que são classificados como ham (não spam) têm um pico mais alto perto do 0.

Algoritmo

Com base nisso podemos construir um algoritmo para prever se um e-mail é spam ou ham. Podemos estimar um modelo onde queremos encontrar uma constante c tal que se a frequência da palavra ”your” for maior que c, então classificamos o e-mail como spam. Caso contrário, classificamos o e-mail como não spam.

Vamos observar graficamente como ficaria esse modelo se c=0.8.

```
plot(density(treino$your[treino$type=="nonspam"]), col="blue",
     main = "Densidade de 'your' em ham (azul) e spam (vermelho)",
     xlab = "Frequência de 'your'", ylab = "densidade")
lines(density(treino$your[treino$type=="spam"]), col="red")
abline(v=0.8,col="black")
```



Os e-mails à direita da linha preta seriam classificados como spam, enquanto que os à esquerda seriam classificados como não spam.

Avaliação

Agora vamos avaliar nosso modelo de predição.

```
predicao=ifelse(treino$your>0.8,"spam","nonspam")
table(predicao,treino$type)/length(treino$type)
```

```
predicao    nonspam      spam
  nonspam 0.4978261 0.1293478
  spam    0.1155797 0.2572464
```

Podemos ver que quando os e-mails não eram spam e classificamos como "não spam", de acordo com nosso modelo, em 50% do tempo nós acertamos. Quando os e-mails eram spam e classificamos ele em spam, por volta de 26% do tempo nós acertamos. Então, ao total, nós acertamos por volta de 50+26=76% do tempo. Então nosso algoritmo de previsão tem uma precisão por volta de 76% na amostra treino.

```
predicao=ifelse(teste$your>0.8,"spam","nonspam")
table(predicao,teste$type)/length(teste$type)
```

```
predicao    nonspam      spam
  nonspam 0.4910375 0.1434003
  spam    0.1037480 0.2618142
```

Já na amostra teste acertamos 48+27=75% das vezes. O erro na amostra teste é o que chamamos de erro real. É o erro que esperamos em amostras novas que passarem por nosso preditor.

Como construir um bom algoritmo de aprendizado de máquina?

O "melhor" método de aprendizado de máquina é caracterizado por:

- Uma boa base de dados;
- Reter informações relevantes;
- Ser bem interpretável;
- Fácil de ser explicado e entendido;
- Ser preciso;

- Fácil de se construir e de se testar em pequenas amostras;
- Fácil aplicar a um grande conjunto de dados.

Os erros mais comuns, que se deve tomar um certo cuidado, são:

- Tentar automatizar a seleção de variáveis (características) de uma maneira que não permita que você entenda como essas variáveis estão sendo aplicadas para fazer previsões;
- Não prestar atenção a peculiaridades específicas de alguns dados, como comportamentos estranhos de variáveis específicas;
- Jogar fora informações desnecessariamente.