Mineração de Dados

Aula 3: Manipulando Textos e Imagens

Rafael Izbicki

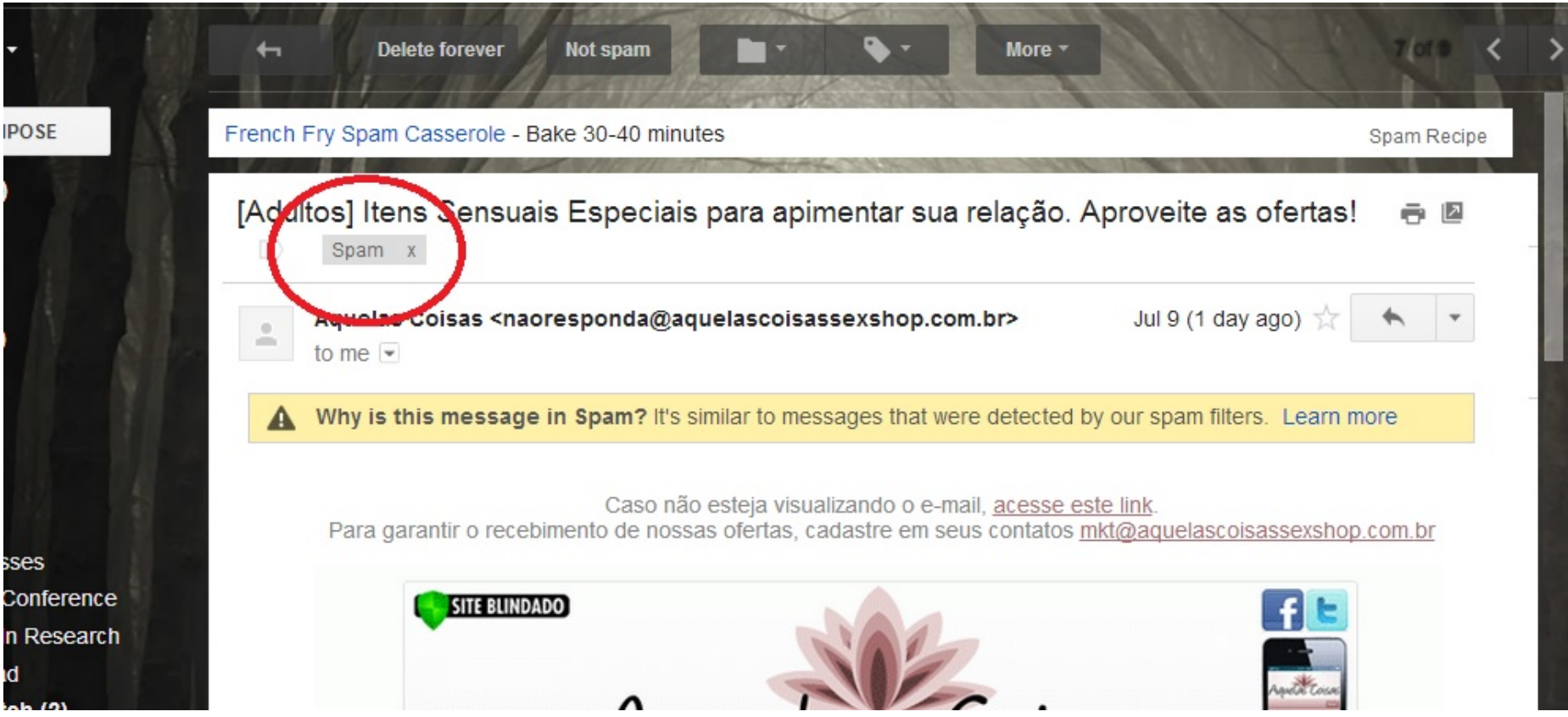
Nesta Aula: Vamos aprender como manipular imagens e textos (no R).

Em estatística, estamos acostumados a trabalhar com objetos x que representam (sequências de) números/vetores.

Ex: Queremos estimar uma função de regressão $\mathbb{E}[Y|x]$

Em mineração de dados, são comuns as aplicações em que x representa objetos não convencionais, como imagens e documentos de texto.

Exemplo: Detecção de Spams



$X_i \longrightarrow$ email

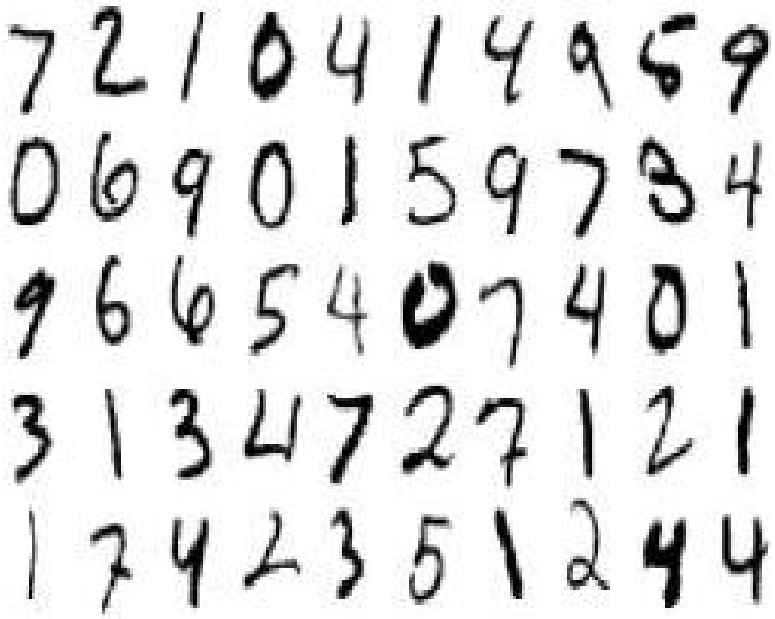
$Y_i \longrightarrow$ spam/não spam

Objetivo: prever Y_i com base em X_i

Exemplo: Information Retrieval

Dados um conjunto de documentos de texto (e.g., páginas de internet), escolher os k documentos mais similares a um dado documento.

Exemplo: Reconhecimento de Dígitos



$X_i \longrightarrow$ imagem de um dígito

$Y_i \longrightarrow$ dígito correspondente

Hoje aprenderemos algumas maneiras básicas de manipular estes objetos

Em uma frase, o que veremos é como converter esses objetos em números.

Tal conversão deve ao mesmo tempo (i) ser rápida de ser feita (ii) ser informativa sobre esses documentos.

Manipulando Texto

Digamos que x é um documento de texto, por exemplo, o texto em uma página de internet, o texto de um tweet, ou de um post do facebook.

Texto 1: $x_1 =$ "Poderoso Estimulante Natural - Esquente sua noite na cama."
Texto 2: $x_2 =$ "Olá professor, sou aluna de Mineração de Dados."
Texto 3: $x_3 =$ "Boa tarde professor Rafael, segue contato como pedido."
Texto 4: $x_4 =$ "Aumente sua performance na cama a noite usando esse estimulante. Esquente seu relacionamento!"

Como convertemos cada um desses texto e um vetor numérico que contenha informação relevante sobre eles?

Bag-of-words – "sacola de palavras"

Muito simples: simplesmente liste as palavras que aparecem nos documentos, e conte quantas vezes elas aparecem.

Texto 1: $x_1 =$"Poderoso Estimulante Natural - Esquente sua noite na cama."
Texto 2: $x_2 =$"Olá professor, sou aluna na aula de Mineração de Dados."
Texto 3: $x_3 =$"Boa tarde professor Rafael, segue contato como pedido."
Texto 4: $x_4 =$"Aumente sua performance na cama a noite usando esse estimulante."

A ideia é fazer uma tabela da seguinte forma:

	poderoso	estimulante	natural	esquente	sua	noite	na	cama	olá	professor	...	esse
Texto 1	1	1	1	1	1	1	1	1	0	0	...	0
Texto 2	0	0	0	0	0	0	1	0	1	1	...	0
Texto 3	0	0	0	0	0	0	0	0	0	1	...	0
Texto 4	0	1	0	1	1	1	1	1	0	0	...	1

Matriz documento-termo

	poderoso	estimulante	natural	esquente	sua	noite	na	cama	olá	professor	...	esse
Texto 1	1	1	1	1	1	1	1	1	0	0	...	0
Texto 2	0	0	0	0	0	0	1	0	1	1	...	0
Texto 3	0	0	0	0	0	0	0	0	0	1	...	0
Texto 4	0	1	0	1	1	1	1	1	0	0	...	1

Note que o vetor relativo ao texto 1 é mais próximo ao vetor relativo ao texto 4 que aos outros. (Distância Euclidiana)

Documentos "próximos" tem distância baixas; documentos "distantes" tem distância alta.

Analogamente, o vetor relativo ao texto 2 é mais próximo ao vetor relativo ao texto 3 que aos outros.

Uma maneira simples de se classificar novos emails (nearest neighbors)

Algumas melhorias adicionais:

Como documentos diferentes tem tamanhos diferentes, pode-se *normalizar* estes vetores (e.g., dividindo-se as frequências absolutas pelo tamanho de cada um dos documentos)

Palavras que são muito comuns muitas vezes não são informativas (e.g., "a", "esse", ...).

Uma maneira de resolver isso é retirar palavras muito comuns. Isso é chato e entendiante

Outra maneira de resolver isso é a chamada **Inverse document frequency (IDF)**, que é mais eficiente.

Seja D o número total de documentos. Para cada palavra w, seja n_w o número de documentos que contém essa palavra. **Para cada vetor de frequências x_d, multiplicamos o w-ésimo elemento por** $\log(D/n_w)$**.**

Para cada vetor de frequências x_d, multiplicamos o w-ésimo elemento por $\log(D/n_w)$.

Intuição: palavras pouco frequentes (n_w baixo) recebem um peso maior.

Ex: A matrix

	poderoso	estimulante	natural	esquente	sua	noite	na	cama	olá	professor	...	esse
Texto 1	1	1	1	1	1	1	1	1	0	0	...	0
Texto 2	0	0	0	0	0	0	1	0	1	1	...	0
Texto 3	0	0	0	0	0	0	0	0	0	1	...	0
Texto 4	0	1	0	1	1	1	1	1	0	0	...	1

passa a ser ($D = 4$)

	poderoso	estimulante	natural	esquente	sua	noite	na	cama	olá	professor	...	esse
Texto 1	2	1	2	1	1	1	0.41	1	0	0	...	0
Texto 2	0	0	0	0	0	0	0.41	0	2	1	...	0
Texto 3	0	0	0	0	0	0	0	0	0	1	...	0
Texto 4	0	1	0	1	1	1	0.41	1	0	0	...	2

É uma espécie de seleção de variáveis

Todas essas são diferentes maneiras de representar um documento de texto. Qual destas é a melhor forma varia conforme a aplicação.

No R: pacote `tm`.

```
dtm = DocumentTermMatrix(corp,
control=list(tolower=TRUE, removePunctuation=TRUE,
removeNumbers=TRUE, stemming=TRUE,
weighting=weightTfIdf))
```

`stemming=TRUE`: palavras com mesma raiz são agrupadas (e.g., connect, connects, connected, connecting)

Nem em todas as línguas isso é tão simples!! Ex: Alemão

Resumo sobre Documentos de Texto:

É comum representarmos um texto por um vetor com as frequências absolutas ou relativas de cada palavra. Esse é o método bag-of-words.

Pode-se multiplicar cada palavra nesta lista por um peso que quantifica o quão comum uma palavra é. A ideia é que palavras muito comuns não são muito informativas.

Uma das formas de se fazer isso é através do Inverse document frequency (IDF).

Manipulando Imagens

Vamos aprender aqui a manipular imagens que tem formato do tipo *raster* (ex: JPEG, PNG, . . .)

raster significa que a imagem é representada por uma ou mais matrizes que contém informações sobre os pixels da figura.

Vamos começar com uma ideia simples: digamos que nós criamos uma matriz binária:

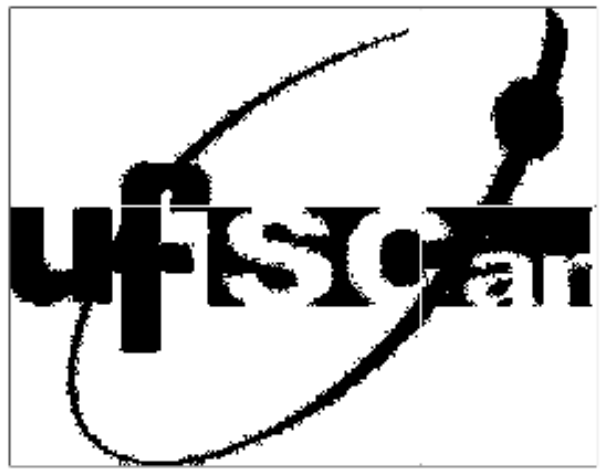
$$\begin{matrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

$$\begin{matrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

A ideia chave é que podemos associar a essa matriz a imagem

Aqui, 1 simboliza um pixel preto, e 0 um pixel branco.

Usando essa ideia, já podemos criar imagens como



Quanto mais pixels, maior a resolução da imagem.

Podemos ir um passo além.

Ao invés de usar apenas 0 (branco) e 1 (preto), podemos usar qualquer número entre 0 e 1 para denotar uma intensidade de cinza.

Com isso, podemos fazer imagens como

Podemos ir ainda mais além.

Lembre-se que usando cores primárias podemos compor qualquer cor.

Usando essa ideia, podemos representar uma imagem com três matrizes simultaneamente.

– A primeira indica quanto azul vamos ter em cada pixel (cada elemento é um número entre 0 e 1)

– A segunda indica quanto amarelo vamos ter em cada pixel (cada elemento é um número entre 0 e 1)

– A terceira indica quanto vermelho vamos ter em cada pixel (cada elemento é um número entre 0 e 1)

Com isso, podemos fazer imagens como

Esse é o princípio usando por exemplo no formato JPEG. Mas, ao invés de usar cores primárias, são usados os RGB channels (vermelho, verde e azul).

Variações sobre o tema:

0 ser branco e 1 preto é apenas uma convenção. Formatos diferente usam convenções diferentes (por exemplo, alguns formatos atribuem 0 a branco e 256 a preto)

Lendo imagens no R

Exemplo artificial:

```
> m=matrix(c(1,1,0,1,1,0,0,0,0),3,3)
> image(m[,3:1],col = c("white","black"))
```

Exemplo do símbolo da ufscar:

```
> library(jpeg)
> imagem=readJPEG("1024px-UFSCar.jpg")
> dim(imagem) [1] 746 1024 3
> image(t(imagem[746:1,,3]),col =
grey.colors(1000,start = 0,end =1)) # imagem em tons
de cinza só com a terceira matriz

> rasterImage(imagem, 0, 0, 1, 1) # imagem colorida
```

Muitas vezes é necessário mudar a resolução de imagens com a finalidade de comparação (i.e., mudar a dimensão das matrizes). Recomendo o MATLAB para isso.

Resumo sobre Imagens:

Imagens nada mais são que matrizes.

Cada elemento da matriz, um pixel, representa a intensidade da cor naquela posição da imagem.

Podemos representar uma imagem com tons de cinza com uma única matriz.

Imagens coloridas costumam ser representadas por 3 matrizes, cada uma indicando a intensidade dos pixels para uma certa cor.