



Formação Inteligência Artificial



Processamento de Linguagem Natural



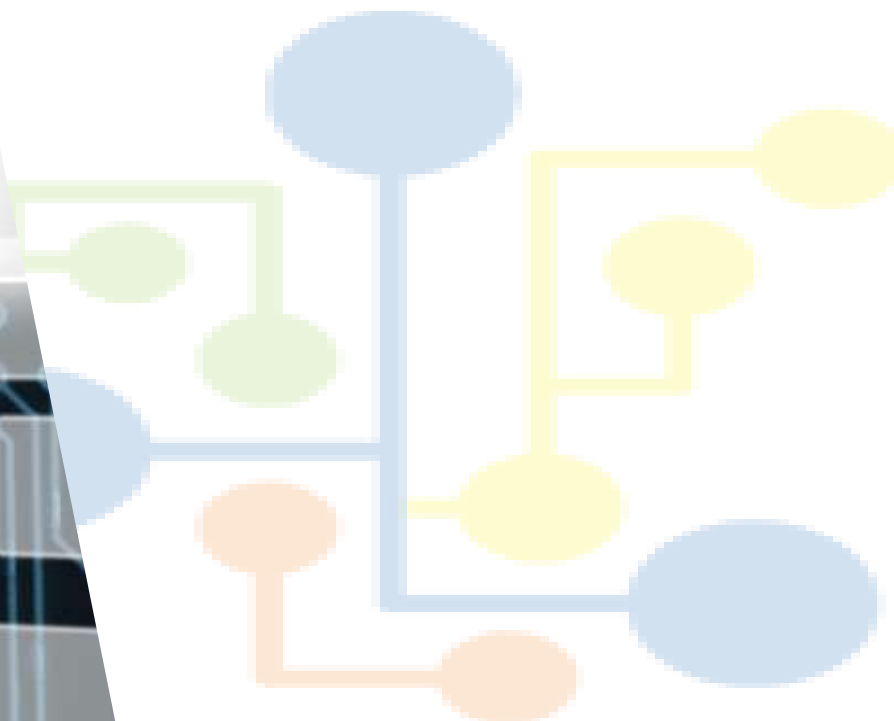


Data Science
Academy

Data Science Academy felipe.oliveiras2000@gmail.com 5f8a0b3ee32fc37d576ba60d

Modelagem Estatística da Linguagem

Parte 1





Modelagem Estatística da Linguagem

Do que precisamos para o Processamento de Linguagem Natural?

- Conhecimento Sobre a Linguagem
- Conhecimento Sobre o Mundo

E como conseguimos alcançar esse objetivo?

- Modelos de Probabilidade para criar modelos de linguagem

$$P(\text{"abrigo"} \rightarrow \text{"casa"})$$



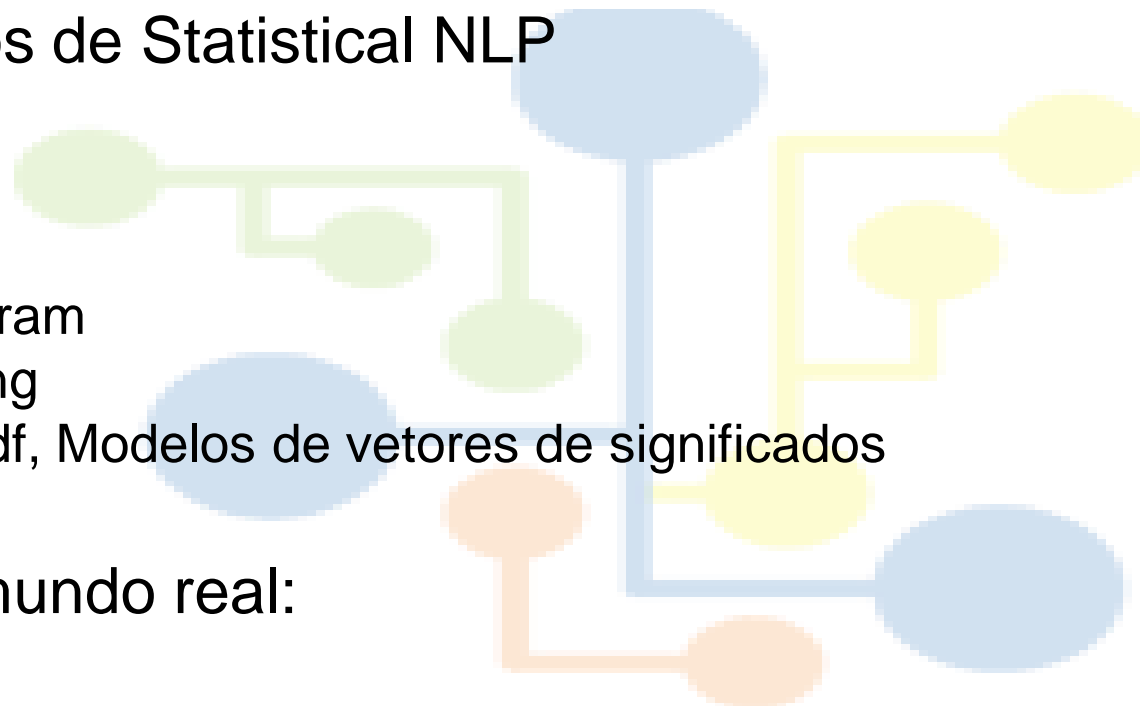
Modelagem Estatística da Linguagem

Teoria e Métodos de Statistical NLP

- Viterbi
- Naïve Bayes
- Modelagem N-gram
- Statistical Parsing
- Invert Index, tf-idf, Modelos de vetores de significados

Aplicações do mundo real:

- Extração de Informação
- Correção de Ortografia
- Recuperação de Informação
- Análise de Sentimentos

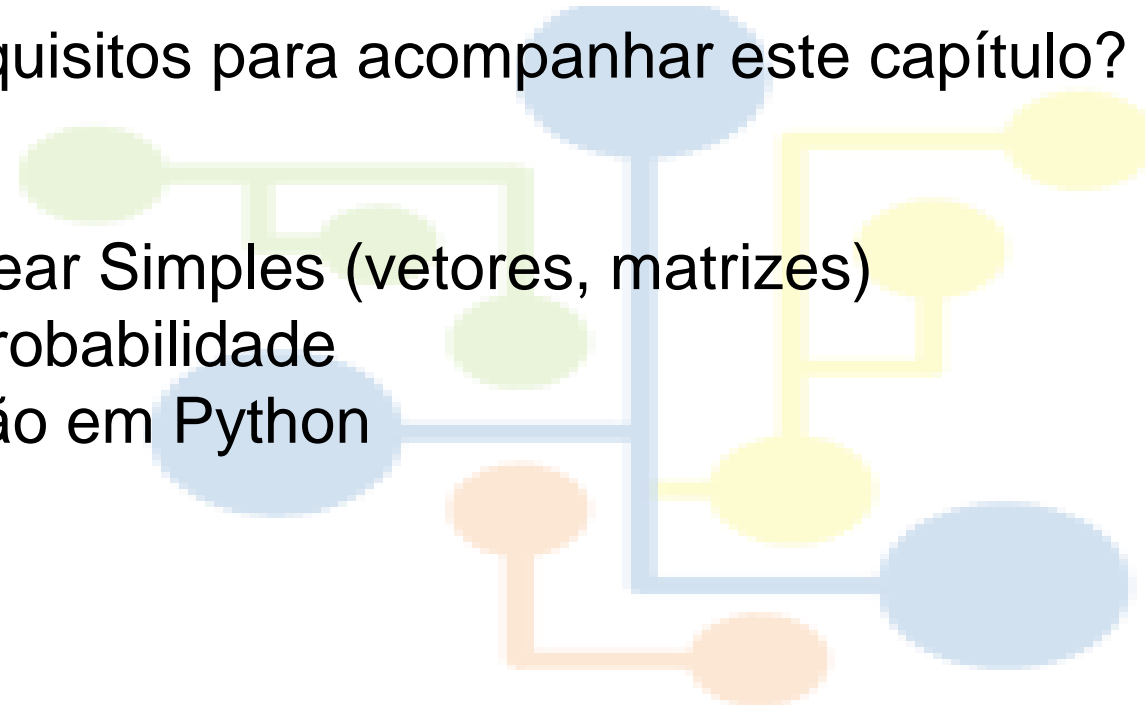




Modelagem Estatística da Linguagem

Quais são os pré-requisitos para acompanhar este capítulo?

- Álgebra Linear Simples (vetores, matrizes)
- Teoria da Probabilidade
- Programação em Python





Normalização de Texto





Normalização de Texto

Cada tarefa de PLN requer:

Segmentação/Tokenização
de palavras

Normalização do Formato

Segmentação de Sentenças



Normalização de Texto

Quantas palavras temos em um texto?

- **Tipo** – um elemento do vocabulário
- **Token** – uma instância de um tipo
- **Vocabulário** – conjunto de tipos

Eles deitaram no gramado do parque em San Francisco e olharam as estrelas.

Tipos = ?

Tokens = ?

Segmentação/Tokenização
de palavras



Normalização de Texto

Quantas palavras temos em um texto?

- **Tipo** – um elemento do vocabulário
- **Token** – uma instância de um tipo
- **Vocabulário** – conjunto de tipos

Eles deitaram no gramado do parque em San Francisco e olharam as estrelas.

Tipos = 12

Tokens = 12

Segmentação/Tokenização
de palavras



Normalização de Texto

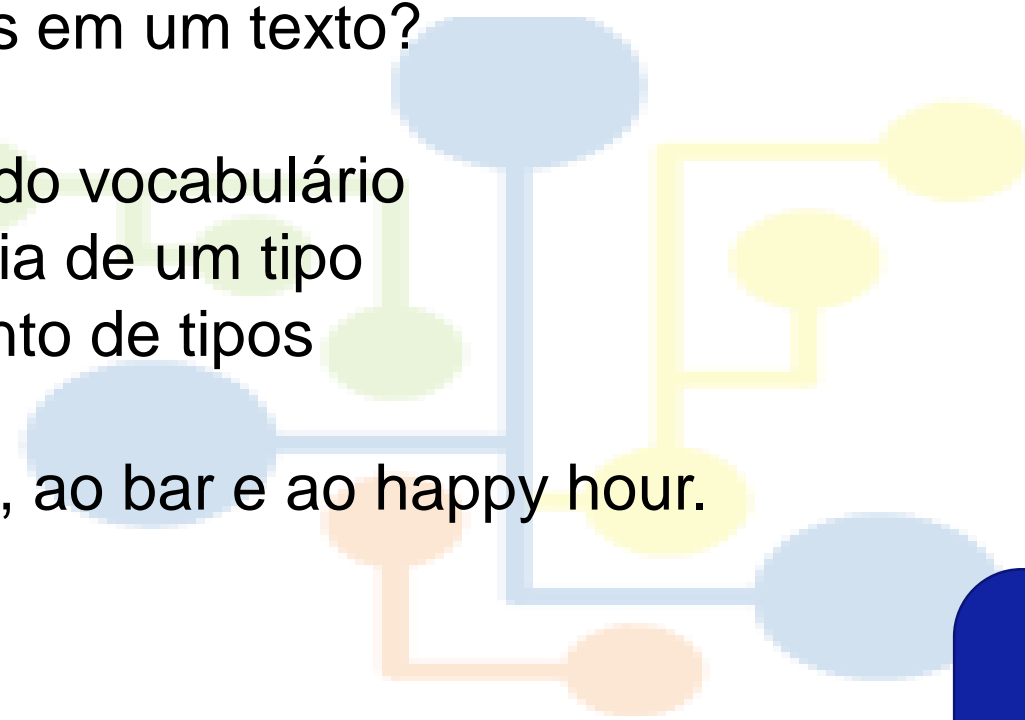
Quantas palavras temos em um texto?

- **Tipo** – um elemento do vocabulário
- **Token** – uma instância de um tipo
- **Vocabulário** – conjunto de tipos

Eles foram ao shopping, ao bar e ao happy hour.

Tipos = ?

Tokens = ?

A decorative background graphic consisting of a network of colored nodes (blue, green, yellow, orange) connected by lines, resembling a tree or a complex graph.

Segmentação/Tokenização
de palavras



Normalização de Texto

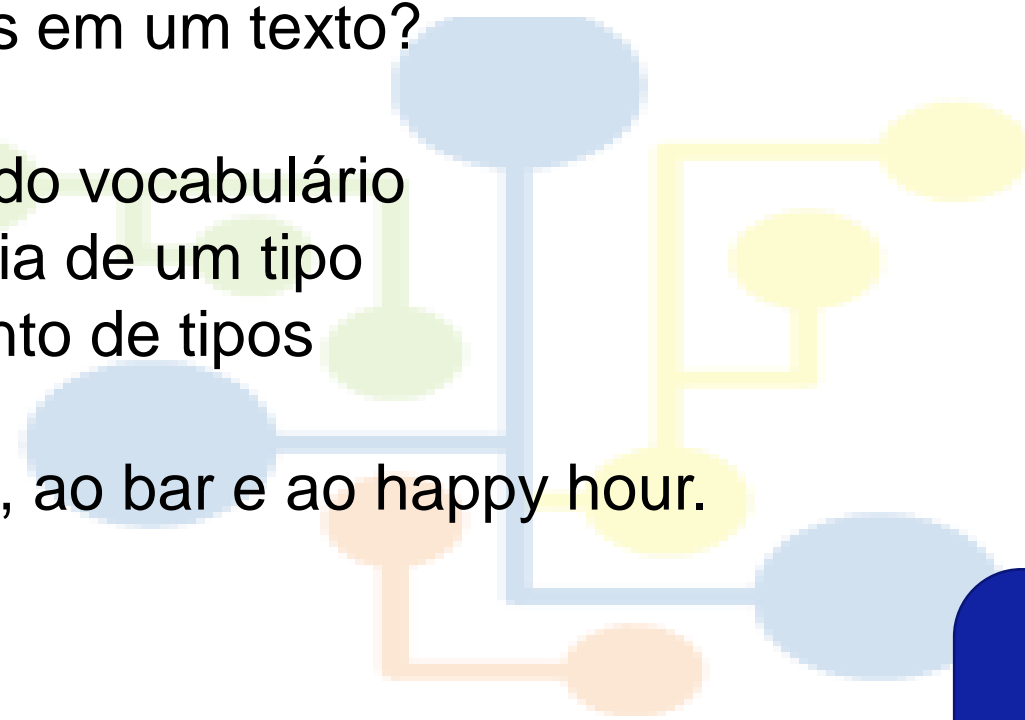
Quantas palavras temos em um texto?

- **Tipo** – um elemento do vocabulário
- **Token** – uma instância de um tipo
- **Vocabulário** – conjunto de tipos

Eles foram ao shopping, ao bar e ao happy hour.

Tipos = 8

Tokens = 10

A decorative background graphic consisting of a network of colored nodes (blue, green, yellow, orange) connected by lines, resembling a stylized tree or a complex graph.

Segmentação/Tokenização
de palavras



Normalização de Texto

Quantas palavras temos em um texto?

- **N = número de tokens**
- **V = vocabulário (conjunto de tipos)**

A decorative background diagram consisting of a central vertical blue line with several horizontal and diagonal branches. At the end of these branches are circles in various colors: light blue, light green, light yellow, and light orange. Some of these circles are further connected to smaller circles of the same color, creating a tree-like structure.

| | Tokens (N) | Tipos (V) |
|---------------|------------|------------|
| Shakespeare | 885.000 | 31.000 |
| Google N-gram | 1 Trilhão | 13 milhões |

Segmentação/Tokenização
de palavras



Normalização de Texto

Normalização:

- Texto indexado e queries precisam ter o mesmo formato.
 - Queremos que U.S.A. seja o mesmo que USA
- Definir classes equivalentes de termos
 - Remover pontos ao final de termos
- Expansão assimétrica
 - Digita: windows → Pesquisa por: window, windows, Windows
 - Digita: Windows → Pesquisa por: Windows

Normalização de Formato



Normalização de Texto

Lemmatization:

- Encontrar a forma correta da palavra no dicionário
- Especialmente importante em Machine Translation
- am, are, is → be
- car, cars, car's, cars' → car

Exemplo: The boy's cars are different colors → the boy car be different color

Lemma → cat e cats = mesmo lemma

Formato da palavra → cat e cats = formatos diferentes

Normalização de Formato



Normalização de Texto

Morfologia

- Morfema – menor unidade com significado que compõe uma palavra
- Stems – a unidade de significado de uma palavra
- Afixos – pedaços ou partes que são anexados aos stems (geralmente com funções gramaticais)

A decorative background graphic consisting of a central blue vertical line with several horizontal and diagonal branches. At the end of these branches are circles in various colors: blue, green, yellow, and orange. The circles vary in size and are arranged in a somewhat symmetrical, tree-like structure.

Normalização de Formato



Normalização de Texto

Stemming:

- Reduz os termos para os stems em recuperação de informação
- Depende da linguagem
- Stemming é uma versão simplificada da lemmatization
- Criamos um Stemmer – conjunto de regras para o Stemmer

Exemplo: automate(s), automatic, automation – reduzidos para automat

Normalização de Formato



Normalização de Texto

Segmentação de Sentenças:

- Limites de sentenças
- O caracter “.” é ambíguo
- Abreviações (o caracter “.” é usado em abreviações)
 - Inc.
 - Dr.
 - 8.5
- Os caracteres ! ou ? não são ambíguos

A decorative background graphic consisting of a central blue vertical line with several horizontal and diagonal branches. At the end of these branches are circles in various colors: blue, green, yellow, and orange. The circles vary in size and are connected by thin lines, creating a network-like structure.

Segmentação de Sentenças



Normalização de Texto

Segmentação de Sentenças:

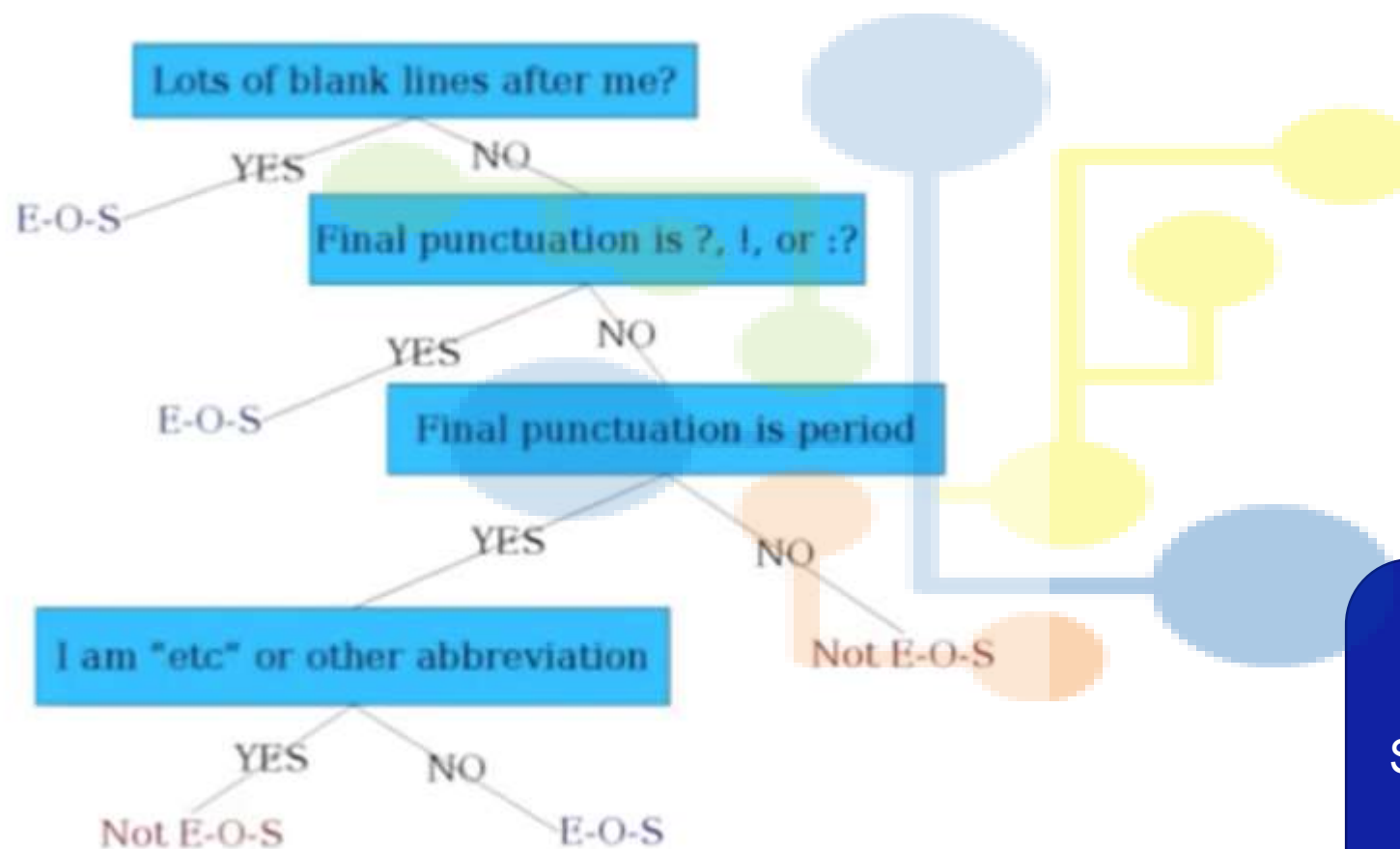
- Construímos um classificador binário
 - Olhamos para “.”
 - Decidimos se é final de sentença ou não
 - Classificadores: regras escritas à mão, expressões regulares ou Machine Learning (árvore de decisão)

A faint, stylized decision tree diagram is visible in the background. It consists of several circular nodes connected by lines. The nodes are colored in light blue, light green, light yellow, and light orange. The lines connecting them are also in these colors, creating a complex web of paths.

Segmentação de Sentenças



Normalização de Texto



Segmentação de Sentenças

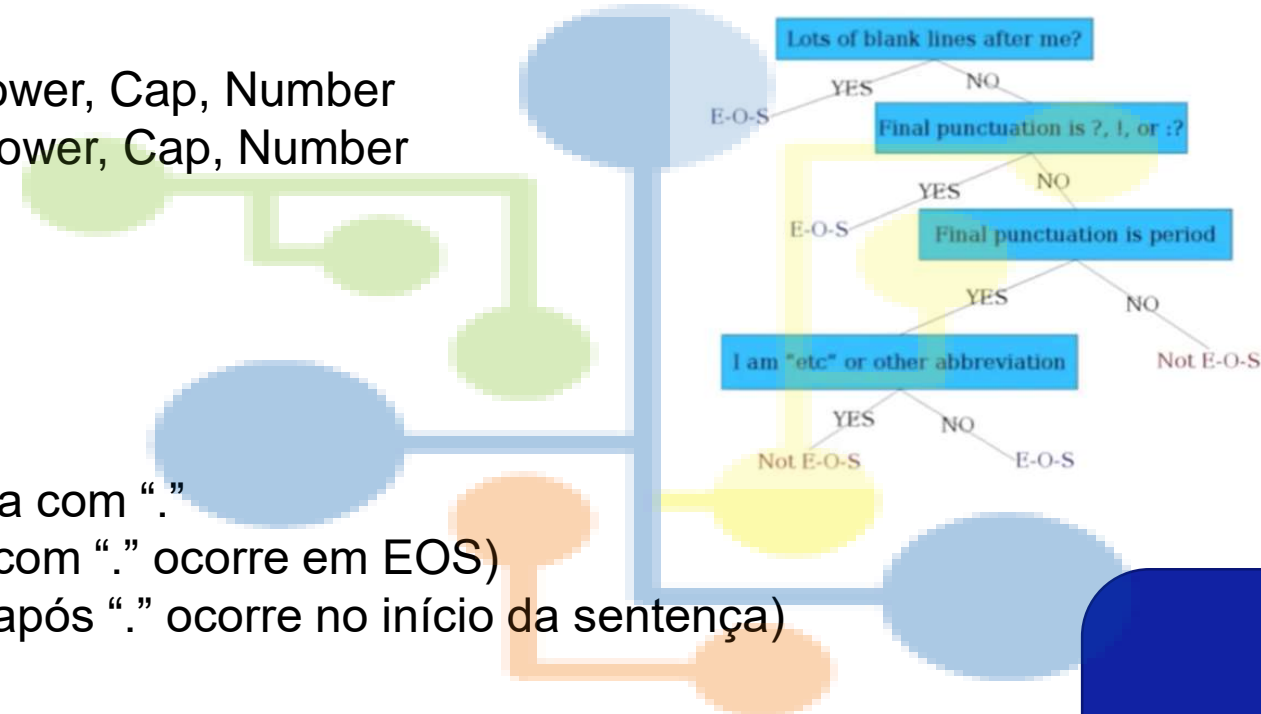


Normalização de Texto

Palavra com "." – Upper, lower, Cap, Number
Palavra após "." – Upper, lower, Cap, Number

Atributos Numéricos:

- Comprimento da palavra com "."
- Probabilidade (palavra com "." ocorre em EOS)
- Probabilidade (palavra após "." ocorre no início da sentença)



Segmentação de Sentenças



Similaridade Entre Strings





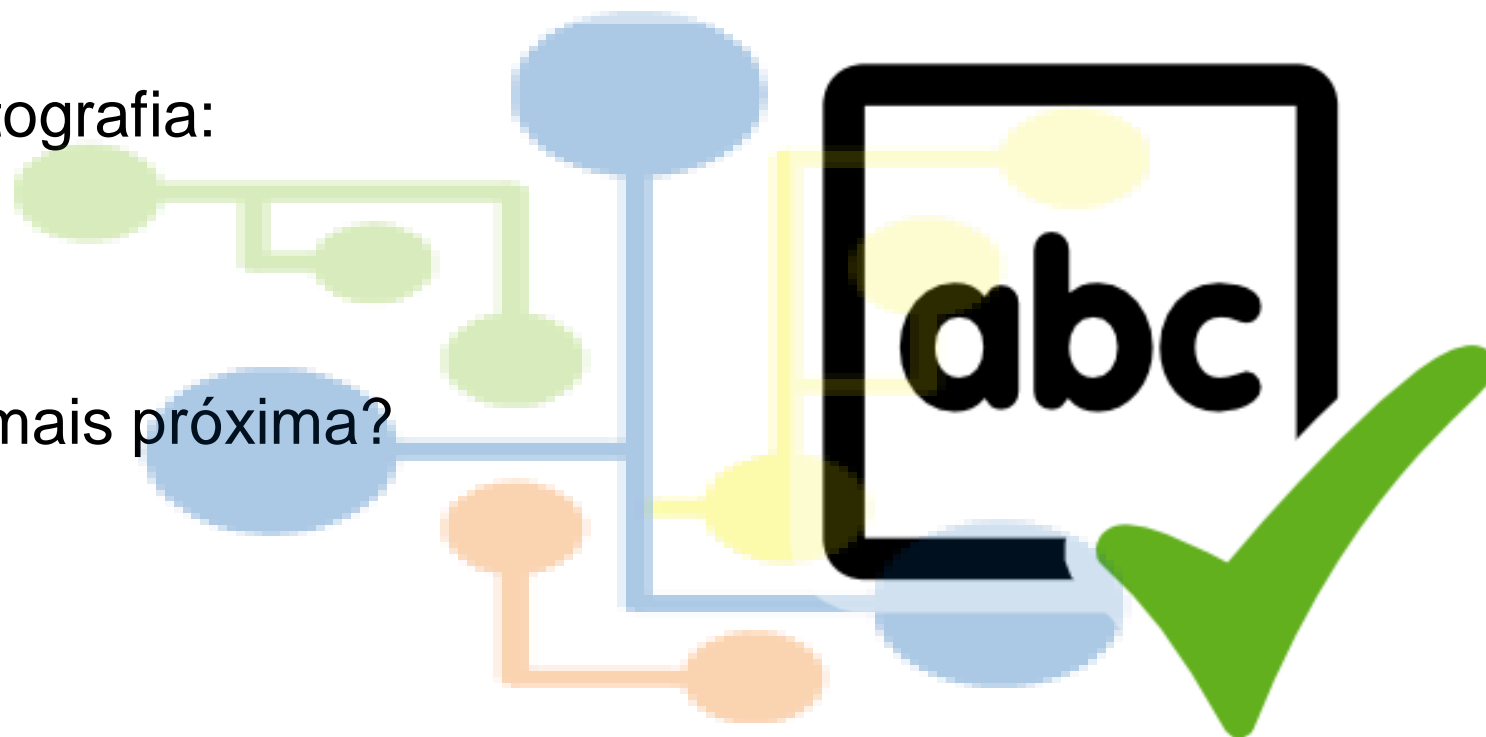
Similaridade Entre Strings

Correção de Ortografia:

“graffe”

Qual a palavra mais próxima?

graf
graft
grail
giraffe





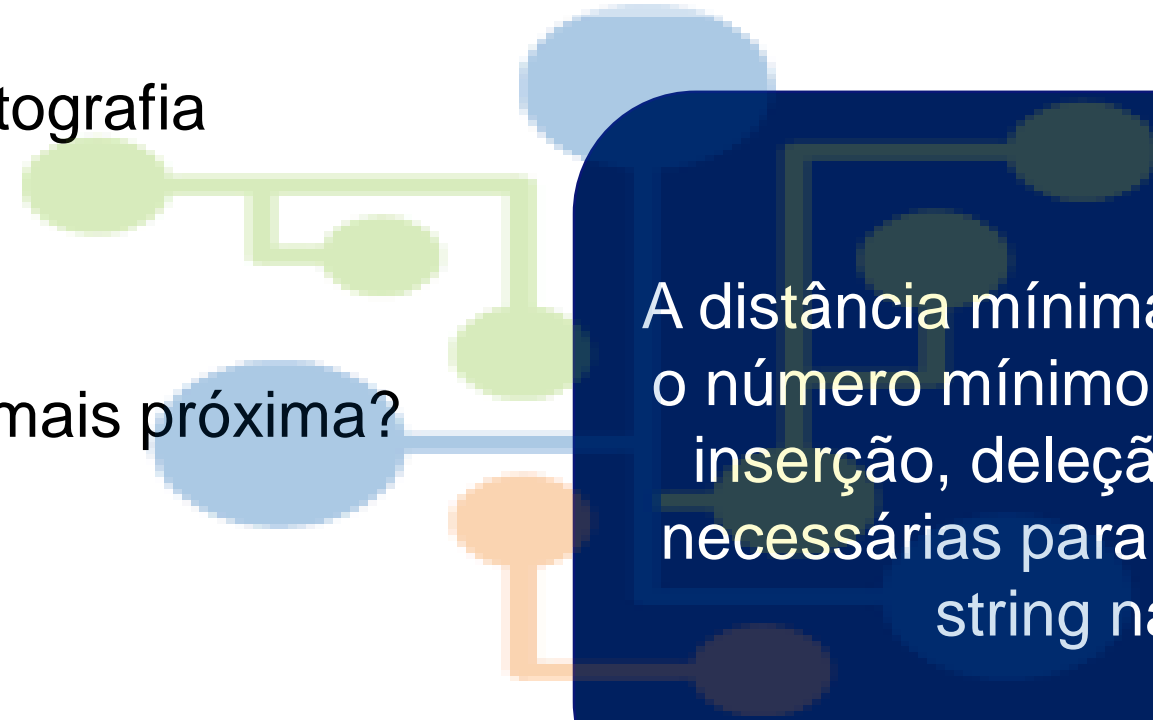
Similaridade Entre Strings

Correção de Ortografia

“graffe”

Qual a palavra mais próxima?

graf
graft
grail
giraffe

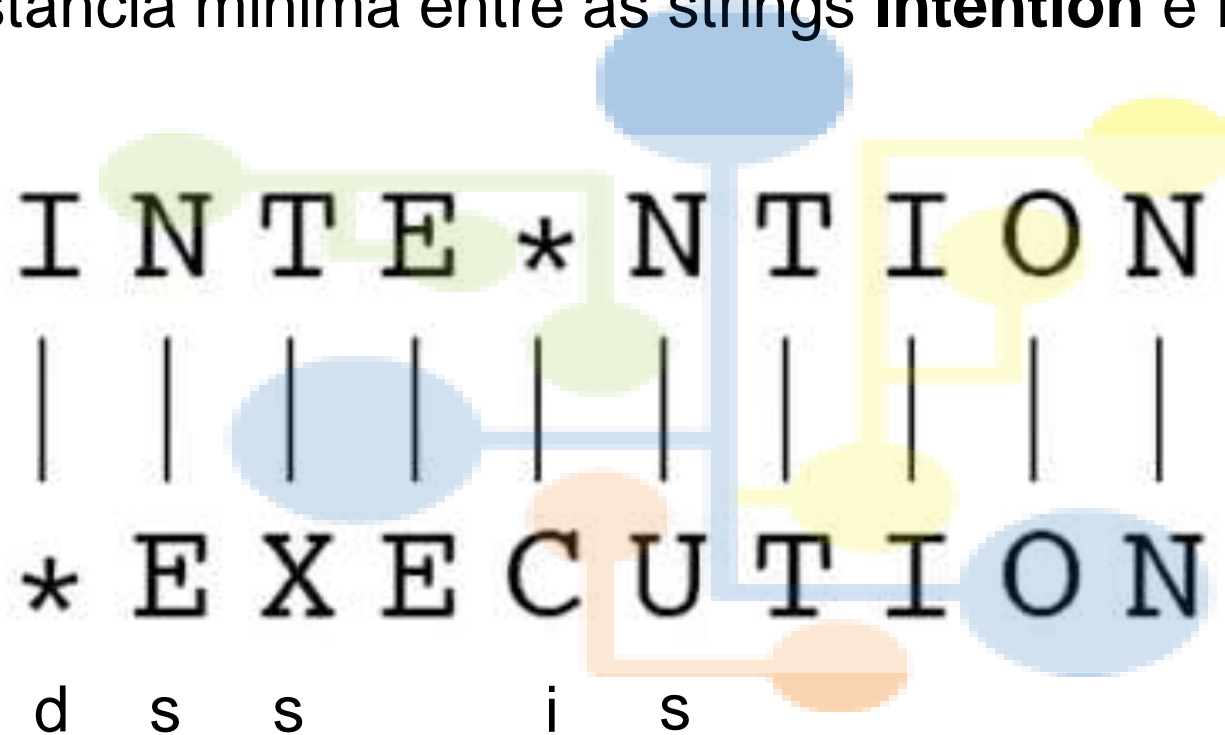
A decorative background graphic consisting of several colored circles (blue, green, orange, dark blue) connected by thin lines, resembling a network or a stylized tree structure.

A distância mínima entre 2 strings é o número mínimo de operações de inserção, deleção e substituição necessárias para transformar uma string na outra.



Similaridade Entre Strings

Qual a distância mínima entre as strings **Intention** e **Execution**?

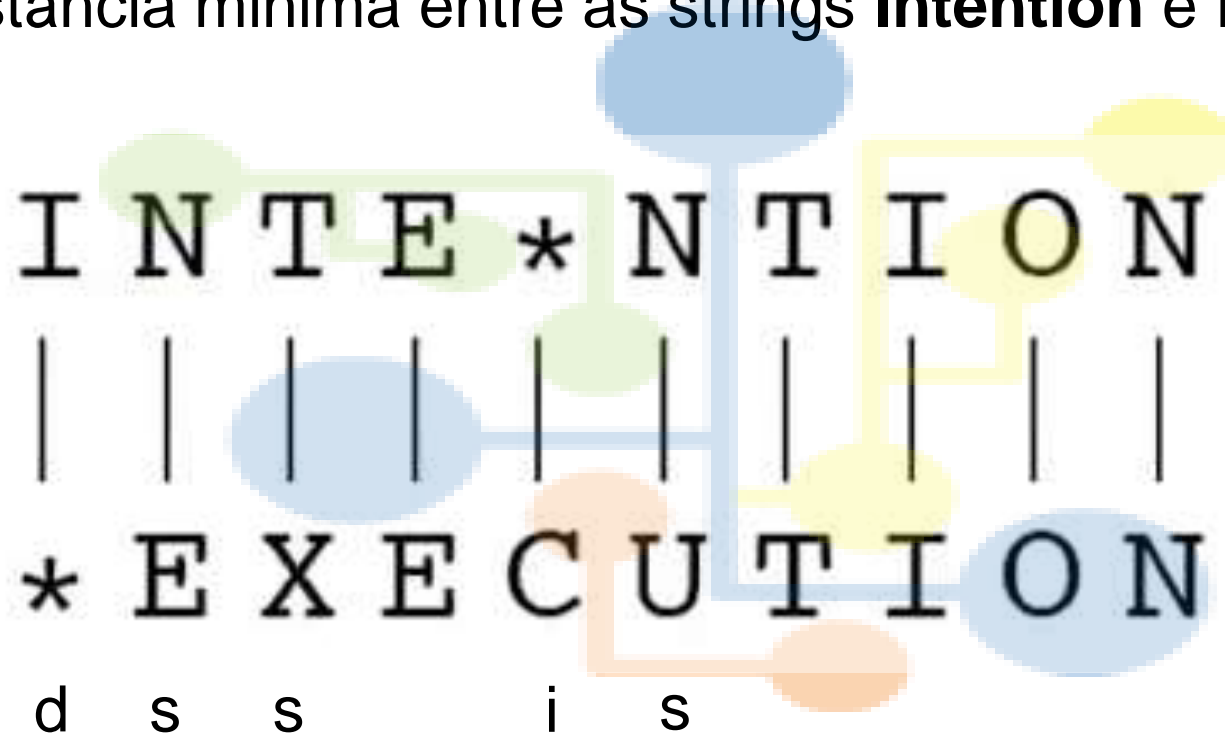


Supondo que cada operação tenha um custo de 1, a distância entre as duas strings seria de 5



Similaridade Entre Strings

Qual a distância mínima entre as strings **Intention** e **Execution**?



Supondo que a operação de substituição tivesse um custo de 2 e as demais operações tivessem um custo de 1, a distância entre as 2 strings seria de 8



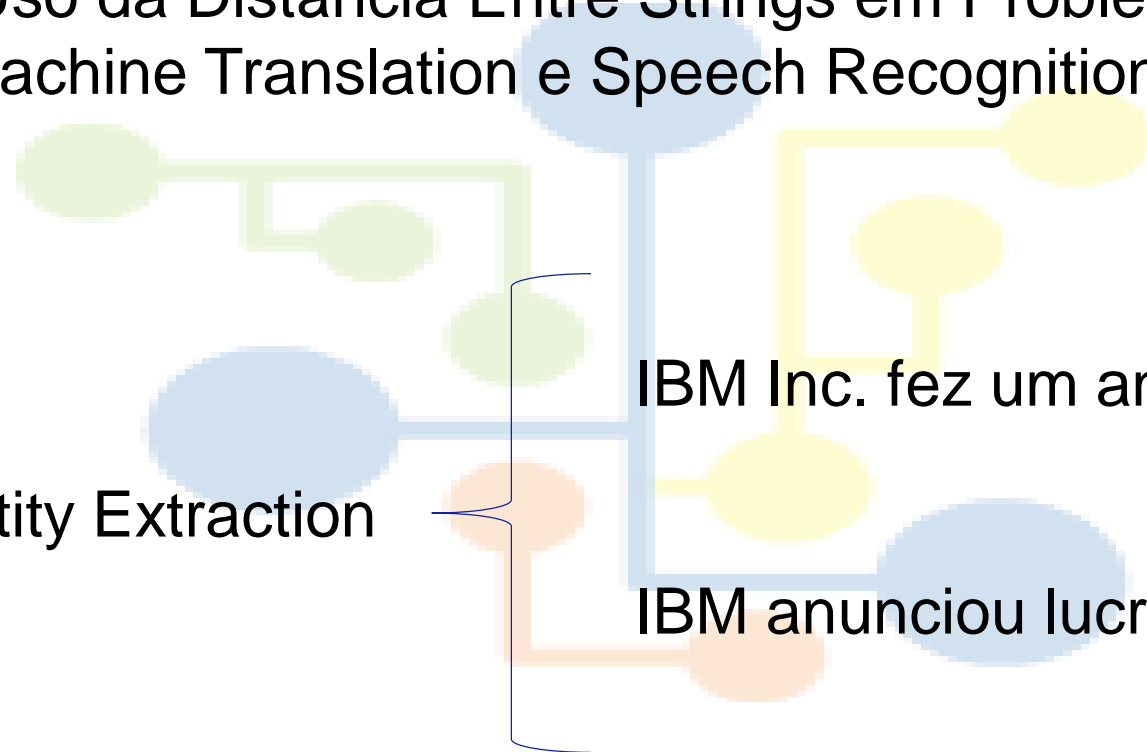
Similaridade Entre Strings

Exemplo de Uso da Distância Entre Strings em Problemas de PLN
Machine Translation e Speech Recognition

Named Entity Extraction

IBM Inc. fez um anúncio hoje

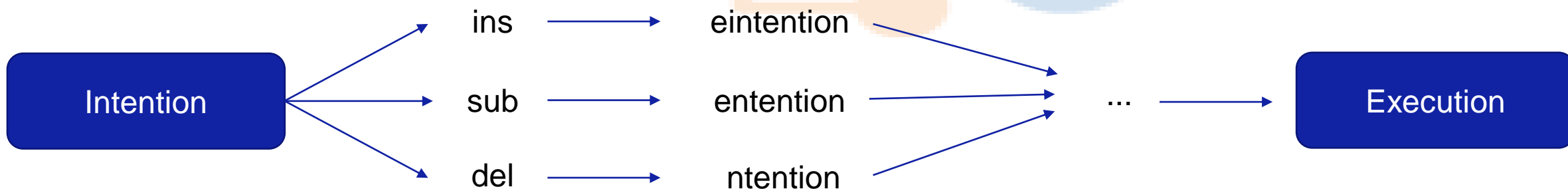
IBM anunciou lucro





Como Encontrar a Distância Mínima?

- Buscamos por um PATH (caminho), que nada mais é do que a sequência de operações de edição.
- Consideramos o PATH da string de início até a string final:
 - **Estado Inicial:** palavra que estamos transformando
 - **Operações:** inserção, deleção, substituição
 - **Estado Objetivo:** palavra que estamos tentando alcançar
 - **Custo do Caminho:** Precisamos minimizar o número de operações de edição.





Como Encontrar a Distância Mínima?

Encontrar a distância entre 2 strings é na verdade um problema de busca!

Estamos em busca do menor caminho entre 2 strings!



Como Encontrar a Distância Mínima?

Para 2 strings:

X de comprimento n
Y de comprimento m

Definimos a distância, como:

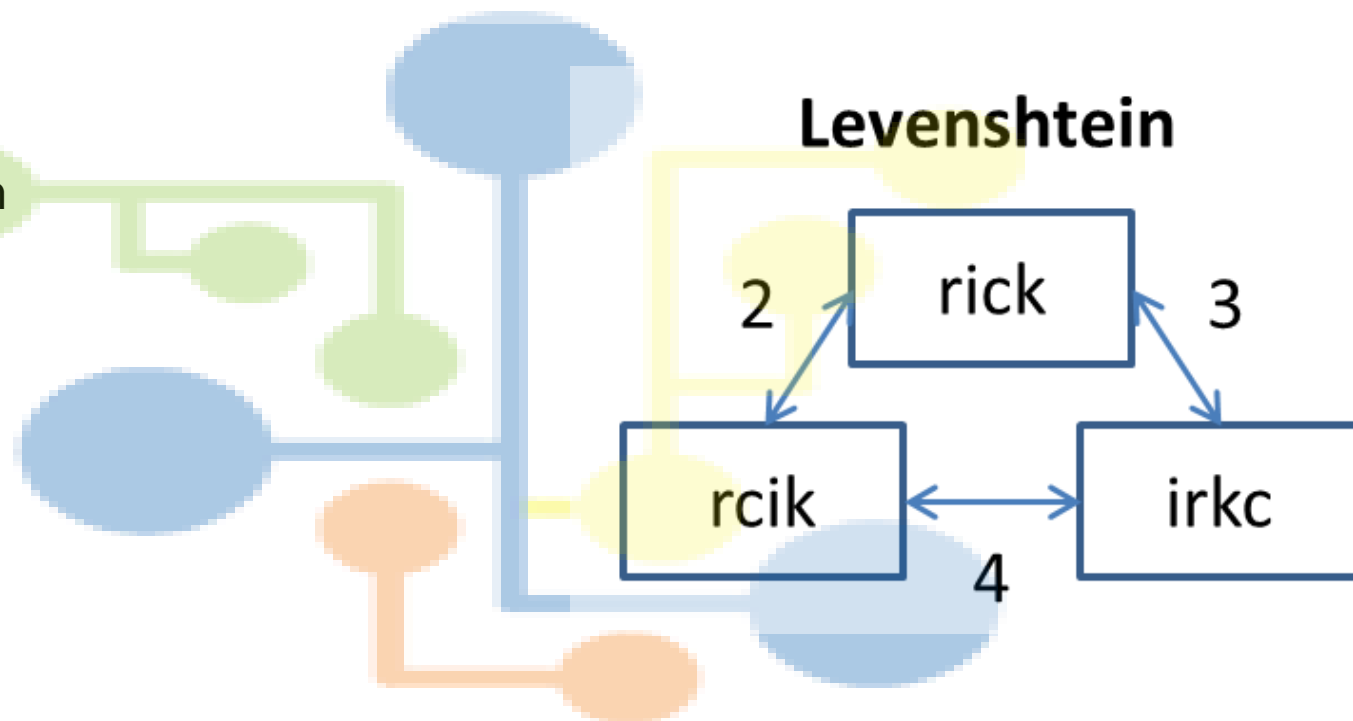
$D(i, j)$

Onde:

$X[1 \dots i]$
 $Y[1 \dots j]$

Logo, a distância entre X e Y, pode ser expressa por:

$D(n, m)$





Como Calcular a Distância Mínima?

- Programação Dinâmica – Computação tabular de $D(n, m)$
- Resolvemos problemas combinando soluções de sub-problemas
 - Computamos $D(i, j)$ para valores pequenos de i e j
 - Computamos valores maiores de $D(i, j)$ com base nos valores menores computados anteriormente
 - Ou seja, computamos $D(i, j)$ para todo valor de i ($0 < i < n$) e j ($0 < j < m$)



Como Calcular a Distância Mínima?

Initialization

$$D(i, 0) = i$$

$$D(0, j) = j$$

Recurrence Relation:

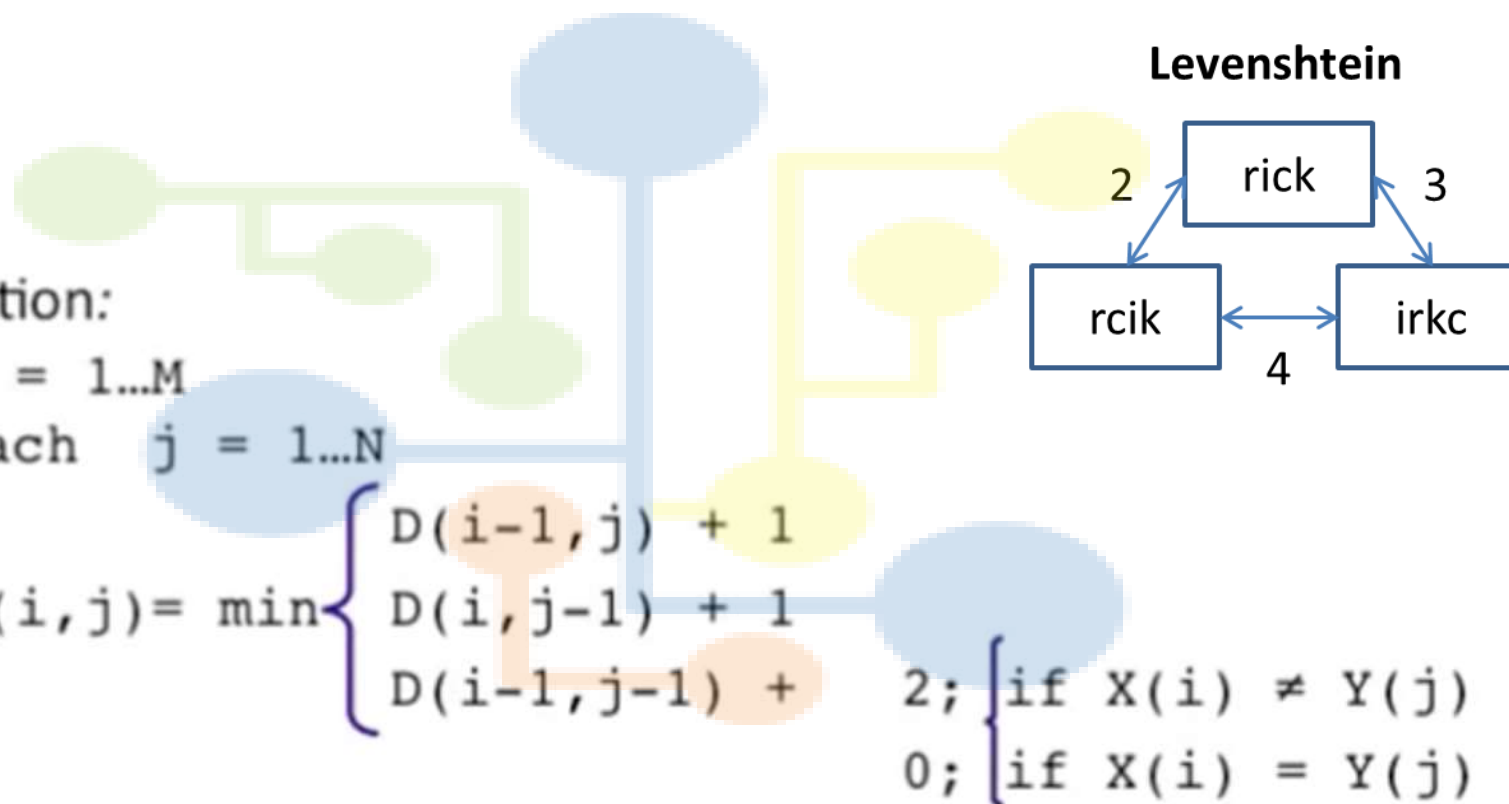
For each $i = 1 \dots M$

For each $j = 1 \dots N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

Termination:

$D(N, M)$ is distance





Como Calcular a Distância Mínima?

Matriz de Distância

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| N | 9 | | | | | | | | | |
| O | 8 | | | | | | | | | |
| I | 7 | | | | | | | | | |
| T | 6 | | | | | | | | | |
| N | 5 | | | | | | | | | |
| E | 4 | | | | | | | | | |
| T | 3 | | | | | | | | | |
| N | 2 | | | | | | | | | |
| I | 1 | | | | | | | | | |
| # | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | # | E | X | E | C | U | T | I | O | N |



Como Calcular a Distância Mínima?

Matriz de Distância

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

| | | | | | | | | | | |
|---|---|---|---|----|----|----|----|----|----|----|
| N | 9 | 8 | 9 | 10 | 11 | 12 | 11 | 10 | 9 | 8 |
| O | 8 | 7 | 8 | 9 | 10 | 11 | 10 | 9 | 8 | 9 |
| I | 7 | 6 | 7 | 8 | 9 | 10 | 9 | 8 | 9 | 10 |
| T | 6 | 5 | 6 | 7 | 8 | 9 | 8 | 9 | 10 | 11 |
| N | 5 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 10 |
| E | 4 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 9 |
| T | 3 | 4 | 5 | 6 | 7 | 8 | 7 | 8 | 9 | 8 |
| N | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 7 | 8 | 7 |
| I | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 6 | 7 | 8 |
| # | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | # | E | X | E | C | U | T | I | O | N |



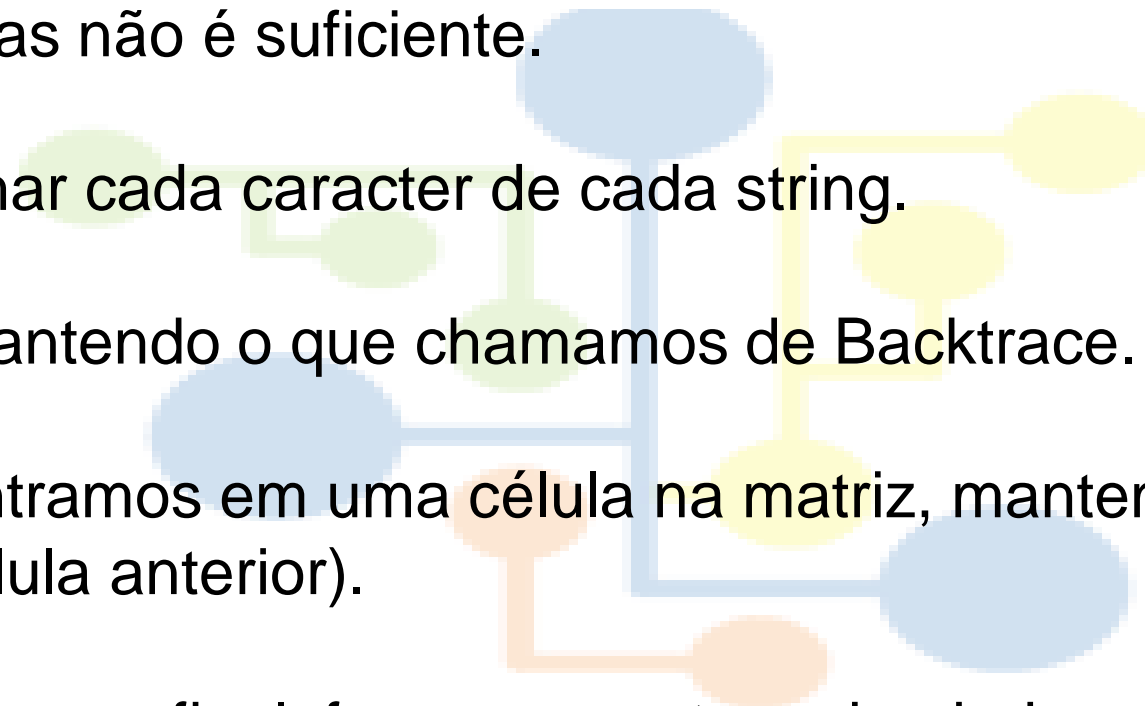
Backtrace





Backtrace

- A distância apenas não é suficiente.
- Precisamos alinhar cada caractere de cada string.
- Fazemos isso mantendo o que chamamos de Backtrace.
- Cada vez que entramos em uma célula na matriz, mantemos informação de onde viemos (célula anterior).
- Quando chegamos ao final, fazemos um trace back do canto superior direito para ler o alinhamento.





Backtrace

| | | | | | | | | | | | |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|--|
| n | 9 | ↓ 8 | ↙←↓ 9 | ↙←↓ 10 | ↙←↓ 11 | ↙←↓ 12 | ↓ 11 | ↓ 10 | ↓ 9 | ↘ 8 | |
| o | 8 | ↓ 7 | ↙←↓ 8 | ↙←↓ 9 | ↙←↓ 10 | ↙←↓ 11 | ↓ 10 | ↓ 9 | ↘ 8 | ← 9 | |
| i | 7 | ↓ 6 | ↙←↓ 7 | ↙←↓ 8 | ↙←↓ 9 | ↙←↓ 10 | ↓ 9 | ↘ 8 | ← 9 | ← 10 | |
| t | 6 | ↓ 5 | ↙←↓ 6 | ↙←↓ 7 | ↙←↓ 8 | ↙←↓ 9 | ↘ 8 | ← 9 | ← 10 | ←↓ 11 | |
| n | 5 | ↓ 4 | ↙←↓ 5 | ↙←↓ 6 | ↙←↓ 7 | ↙←↓ 8 | ↙←↓ 9 | ↙←↓ 10 | ↙←↓ 11 | ↙↓ 10 | |
| e | 4 | ↘ 3 | ← 4 | ↙← 5 | ← 6 | ← 7 | ←↓ 8 | ↙←↓ 9 | ↙←↓ 10 | ↓ 9 | |
| t | 3 | ↙←↓ 4 | ↙←↓ 5 | ↙←↓ 6 | ↙←↓ 7 | ↙←↓ 8 | ↙ 7 | ←↓ 8 | ↙←↓ 9 | ↓ 8 | |
| n | 2 | ↙←↓ 3 | ↙←↓ 4 | ↙←↓ 5 | ↙←↓ 6 | ↙←↓ 7 | ↙←↓ 8 | ↓ 7 | ↙←↓ 8 | ↙ 7 | |
| i | 1 | ↙←↓ 2 | ↙←↓ 3 | ↙←↓ 4 | ↙←↓ 5 | ↙←↓ 6 | ↙←↓ 7 | ↙ 6 | ← 7 | ← 8 | |
| # | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | |
| | # | e | x | e | c | u | t | i | o | n | |



Backtrace

Base conditions:

$$D(i, 0) = i$$

$$D(0, j) = j$$

Termination:

$D(N, M)$ is distance

Recurrence Relation:

For each $i = 1 \dots M$

For each $j = 1 \dots N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

$$\text{ptr}(i, j) = \begin{cases} \text{LEFT} \\ \text{DOWN} \\ \text{DIAG} \end{cases}$$

deletion

insertion

substitution

insertion

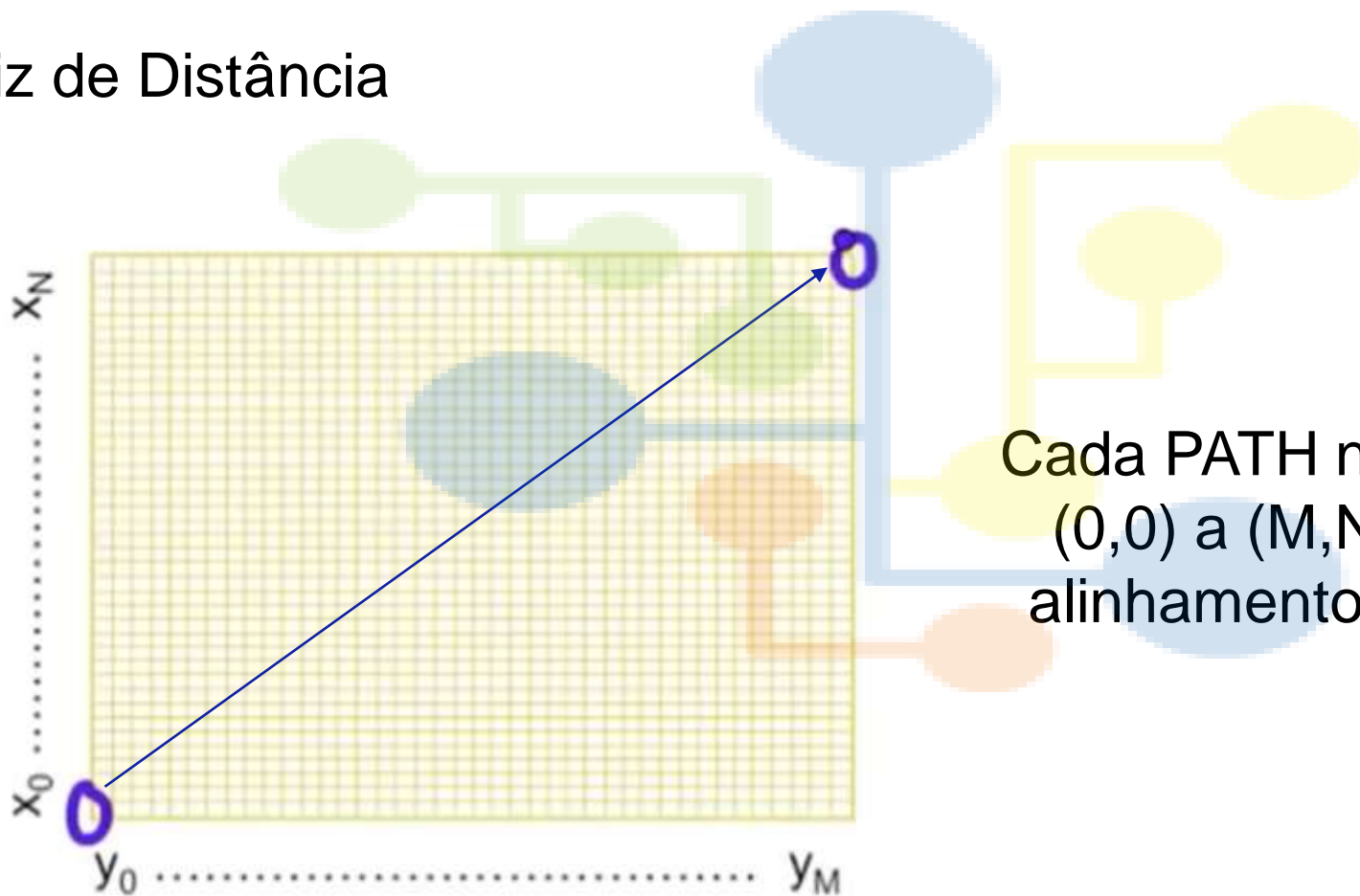
deletion

substitution



Backtrace

Matriz de Distância



Cada PATH não-descendente de $(0,0)$ a (M,N) corresponde ao alinhamento entre duas strings



Distância Mínima Ponderada

Por que adicionamos pesos na computação da distância?

Correção Ortográfica – algumas letras são normalmente digitadas de forma errada com maior frequência que outras.





Distância Mínima Ponderada

Initialization:

$$D(0,0) = 0$$

$$D(i,0) = D(i-1,0) + \text{del}[x(i)]; \quad 1 < i \leq N$$

$$D(0,j) = D(0,j-1) + \text{ins}[y(j)]; \quad 1 < j \leq M$$

Recurrence Relation:

$$D(i,j) = \min \begin{cases} D(i-1,j) + \text{del}[x(i)] \\ D(i,j-1) + \text{ins}[y(j)] \\ D(i-1,j-1) + \text{sub}[x(i),y(j)] \end{cases}$$

Termination:

$D(N,M)$ is distance



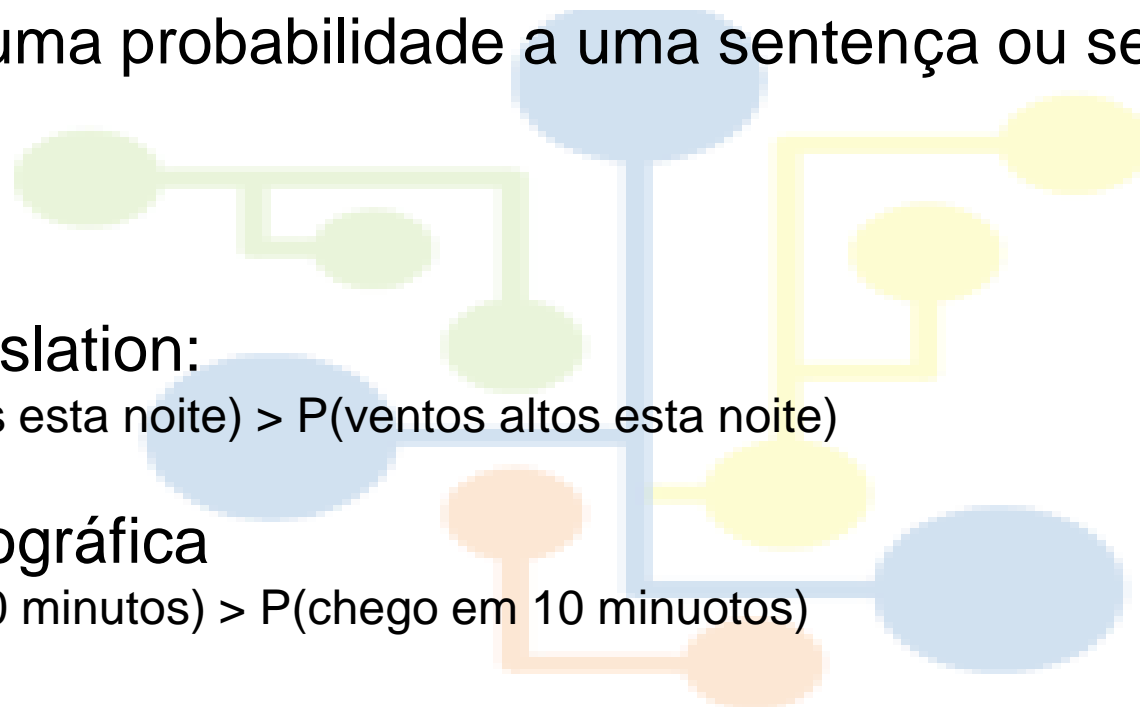
Modelagem Probabilística





Modelagem Probabilística

- Objetivo: Atribuir uma probabilidade a uma sentença ou sequência de palavras.
- Por que?
 - Machine Translation:
 - $P(\text{ventos fortes esta noite}) > P(\text{ventos altos esta noite})$
 - Correção Ortográfica
 - $P(\text{chego em 10 minutos}) > P(\text{chego em 10 minuotos})$
 - Reconhecimento de Voz
 - $P(\text{Eu vi a lua}) > P(\text{Euvira lua})$





Modelagem Probabilística

- Objetivo: Atribuir uma probabilidade a uma sentença ou sequência de palavras.

A decorative background featuring several overlapping circles in light blue, light green, and light yellow, connected by thin lines of the same colors.
$$\Pr(W) = \Pr(w_1, w_2, \dots, w_n)$$

- Probabilidade da próxima palavra (predição de texto).

$$\Pr(w_5 \mid w_4, w_3, w_2, w_1)$$

- O modelo que computa essas probabilidades é chamado de modelo de linguagem, embora às vezes seja chamado de gramática.



Chain Rule

- Como calculamos essa probabilidade?
- Regra do produto (Chain Rule)
 - Como calcular: $\Pr(o, \text{tr\~{a}nsito, estava, lento, pr\~{o}ximo})$?
 - Intuiç~{a}o: considerar a regra do produto de probabilidades (Probabilidade condicional)
 - $\Pr(A,B,C,D) = \Pr(A) \cdot \Pr(B|A) \cdot \Pr(C|A,B) \cdot \Pr(D|A,B,C)$
 - A: inteligente, B: rico, C: bonito e D: solteiro.



Chain Rule

- Como calculamos essa probabilidade?

$$\begin{aligned} \Pr(o, \text{tr\~{a}nsito}, \text{estava}, \text{lento}, \text{pr\'oximo}) &= \Pr(o) \\ &\cdot \Pr(\text{tr\~{a}nsito} | o) \\ &\cdot \Pr(\text{estava} | \text{tr\~{a}nsito}, o) \\ &\cdot \Pr(\text{lento} | \text{estava}, \dots, o) \\ &\cdot \Pr(\text{pr\'oximo} | \text{lento}, \dots, o). \end{aligned}$$

- Mas como estimar tais probabilidades? Solução: contar e dividir.

$$\Pr(\text{pr\'oximo} | \text{lento}, \text{estava}, \text{tr\~{a}nsito}, o) = \text{count}(\text{pr\'oximo}, \text{lento}, \text{estava}, \text{tr\~{a}nsito}, o) / \text{count}(\text{lento}, \text{estava}, \text{tr\~{a}nsito}, o)$$

- Porém, não temos como aplicar esta solução, pois não teríamos dados suficientes para isso.



Chain Rule

E como resolvemos isso, afinal?

Usando a Suposição de Markov
(ou Markov Assumption)



Suposição de Markov

- Usa-se a suposição simplificada de Markov
 - $\Pr(\text{próximo}|\text{lento},\text{estava},\text{trânsito},\text{o}) \approx \Pr(\text{próximo}|\text{lento})$
- Ou talvez
 - $\Pr(\text{próximo}|\text{lento},\text{estava},\text{trânsito},\text{o}) \approx \Pr(\text{próximo}|\text{lento},\text{estava})$
- Dessa forma, aproxima-se as conjuntas por produtos de poucos termos:

$$P(w_1 w_2 \dots w_n) \approx \prod_i P(w_i | w_{i-k} \dots w_{i-1})$$

$$\Pr(w_i | w_{i-1}, \dots, w_1) \approx \Pr(w_i | w_{i-1}, \dots, w_{i-k})$$



Modelos N-gramas

- O modelo mais simples é o unigrama:

$$\Pr(w_1, w_2, \dots, w_n) = \prod_{i=1}^n \Pr(w_i)$$

- O bi-grama usa k=2

$$\Pr(w_i | w_{i-1}, \dots, w_1) = \Pr(w_i | w_{i-1})$$

- A ideia pode ser expandida para tri-grama, 4-grama, etc. Em geral, esse é um modelo de linguagem insuficiente. Porém, é útil. A linguagem tem dependências de longa distância.



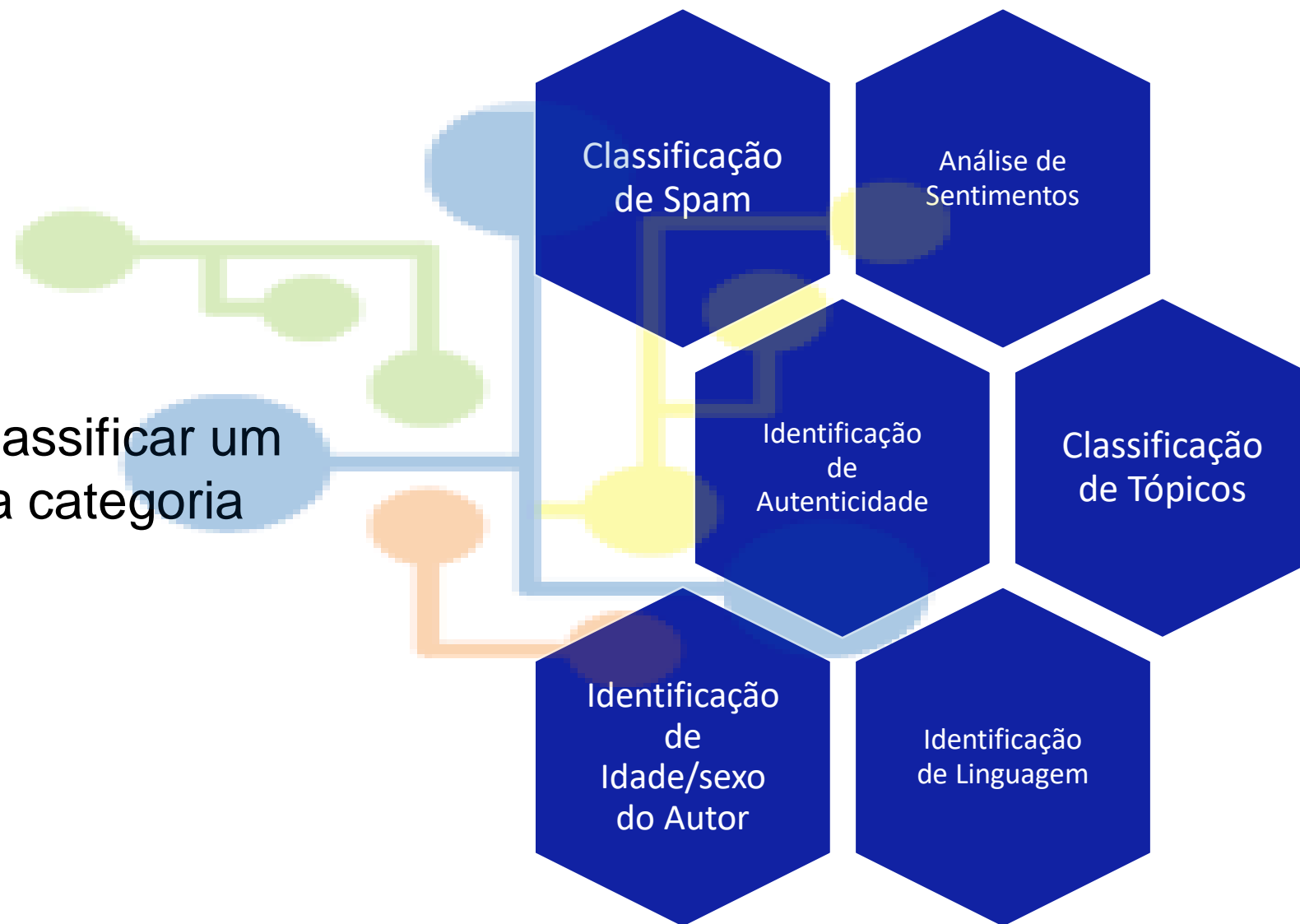
Classificação de Texto





Classificação de Texto

Processo de classificar um texto em uma categoria





Classificação de Texto

O Processo de Classificação de Texto:

Input

Um documento d

Um conjunto de classes $C = \{c_1, c_2, \dots, c_n\}$

Output

Classe prevista
 c deve pertencer a C



Classificação de Texto

Métodos de Classificação de Texto

Método baseado em regras (definidas manualmente por um expert)

Aprendizagem supervisionada



Classificação de Texto

Métodos de Classificação de Texto

- Input

- Um documento d
- Um conjunto de classes C
- Um dataset de treino $(d_1, c_1), \dots, (d_n, c_n)$

- Output

- Classificador treinado $y: d \rightarrow c$

Aprendizagem supervisionada



Classificação de Texto

Métodos de Classificação de Texto

- Naive Bayes
- Regressão Logística
- Support Vector Machines
- KNN
- Redes Neurais Artificiais
- Deep Learning

Aprendizagem supervisionada



Representação Bag of Words





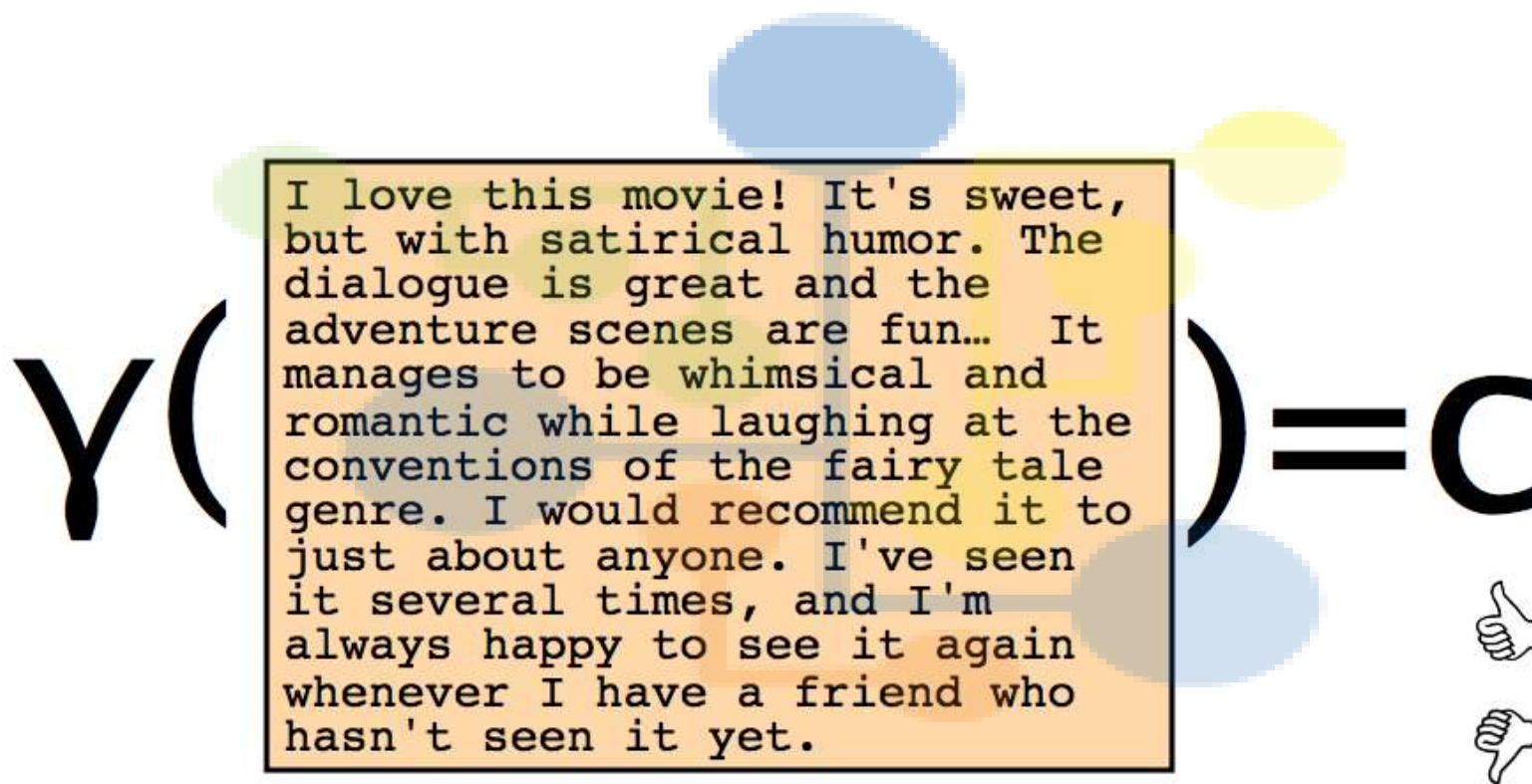
Representação Bag of Words

Naïve Bayes é um método de classificação baseado no Teorema de Bayes.

O Naïve Bayes considera uma representação simples de um documento, chamada Bag of Words.

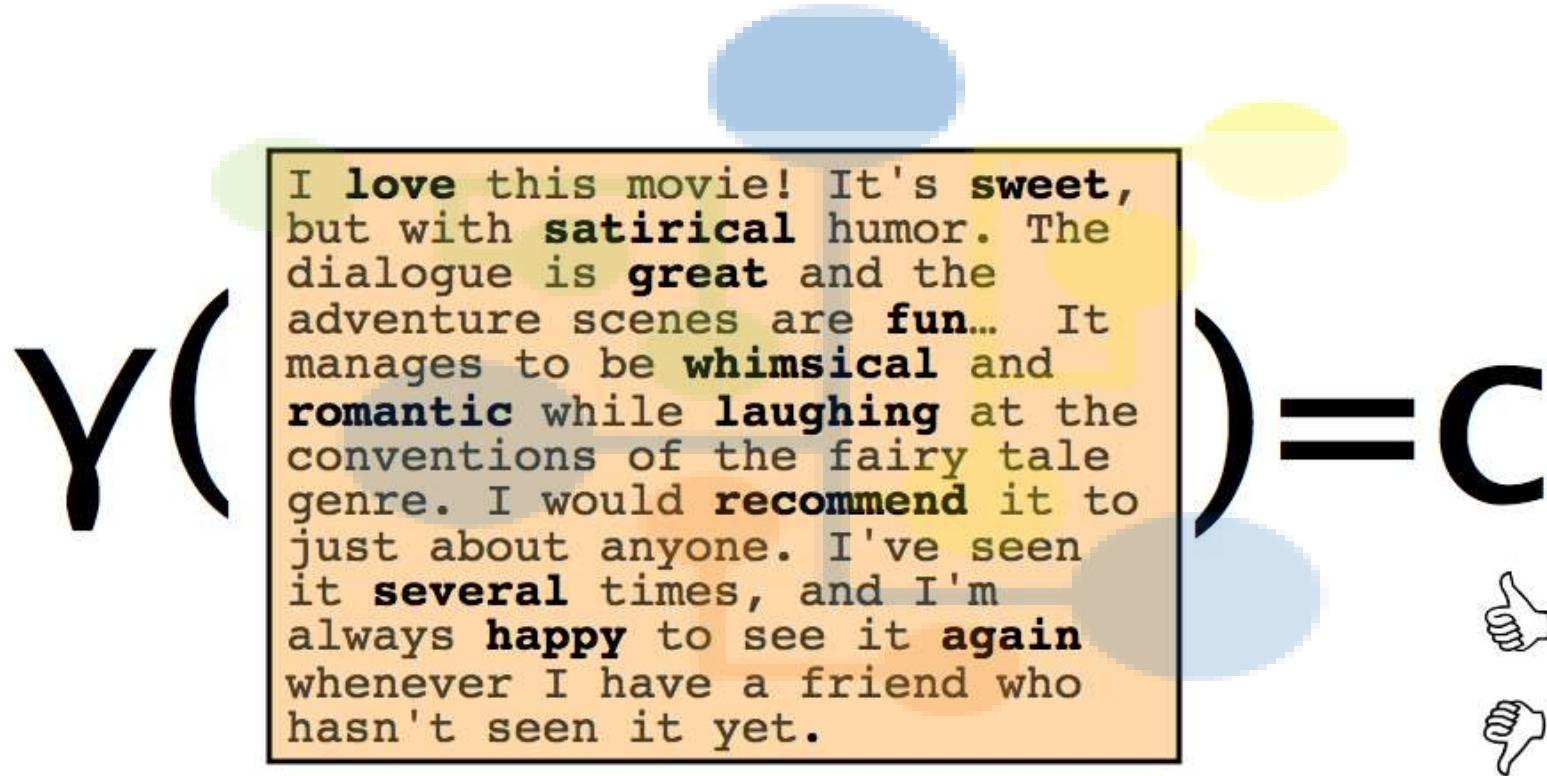


Representação Bag of Words





Representação Bag of Words



Representação Bag of Words





Representação Bag of Words

$Y(\text{great love recommend laugh happy} \dots) = C$

| | |
|-----------|-----|
| great | 2 |
| love | 2 |
| recommend | 1 |
| laugh | 1 |
| happy | 1 |
| ... | ... |

👍
👎



Classificador Naïve Bayes





Classificador Naïve Bayes

Teorema de Bayes

A regra de Bayes mostra como alterar as probabilidades a priori tendo em conta novas evidências de forma a obter probabilidades a posteriori.

$$\Pr(A|B) = \frac{\Pr(B|A) \Pr(A)}{\Pr(B)}$$

$\Pr(A)$ e $\Pr(B)$ são as probabilidades a priori de A e B.

$\Pr(B|A)$ e $\Pr(A|B)$ são as probabilidades a posteriori de B condicional a A e de A condicional a B respectivamente.



Classificador Naïve Bayes

Teorema de Bayes

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

Probabilidade

Probabilidade original da Classe

Probabilidade posterior

Preditor da probabilidade posterior

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

- $P(c | x)$ é a probabilidade posterior da classe (c, alvo) dada o preditor (x, atributos).
- $P(c)$ é a probabilidade original da classe.
- $P(x | c)$ é a probabilidade do preditor dada a classe.
- $P(x)$ é a probabilidade original do preditor.



Classificador Naïve Bayes

Teorema de Bayes

Em teoria da probabilidade o Teorema de Bayes mostra a relação entre uma probabilidade condicional e a sua inversa.



Classificador Naïve Bayes

Para um documento d e uma classe c , temos:

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



Classificador Naïve Bayes

Para um documento d e uma classe c , temos:

$$C_{MAP} = \operatorname{argmax}_{c \in C} P(c | d)$$

MAP = "Maximum a Posteriori" ou classe mais provável

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)}$$

Regra de Bayes

$$= \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

Removemos o denominador, pois $P(d)$ será a mesma para todas as classes

Probabilidade

Priori



Classificador Naïve Bayes

Para um documento d e uma classe c , temos:

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(d | c) P(c)$$

$$= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c) P(c)$$

Documento d é representado
pelas features x_1, x_2, \dots, x_n



Classificador Naïve Bayes

Para um documento d e uma classe c , temos:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \dots, x_n | c) P(c)$$

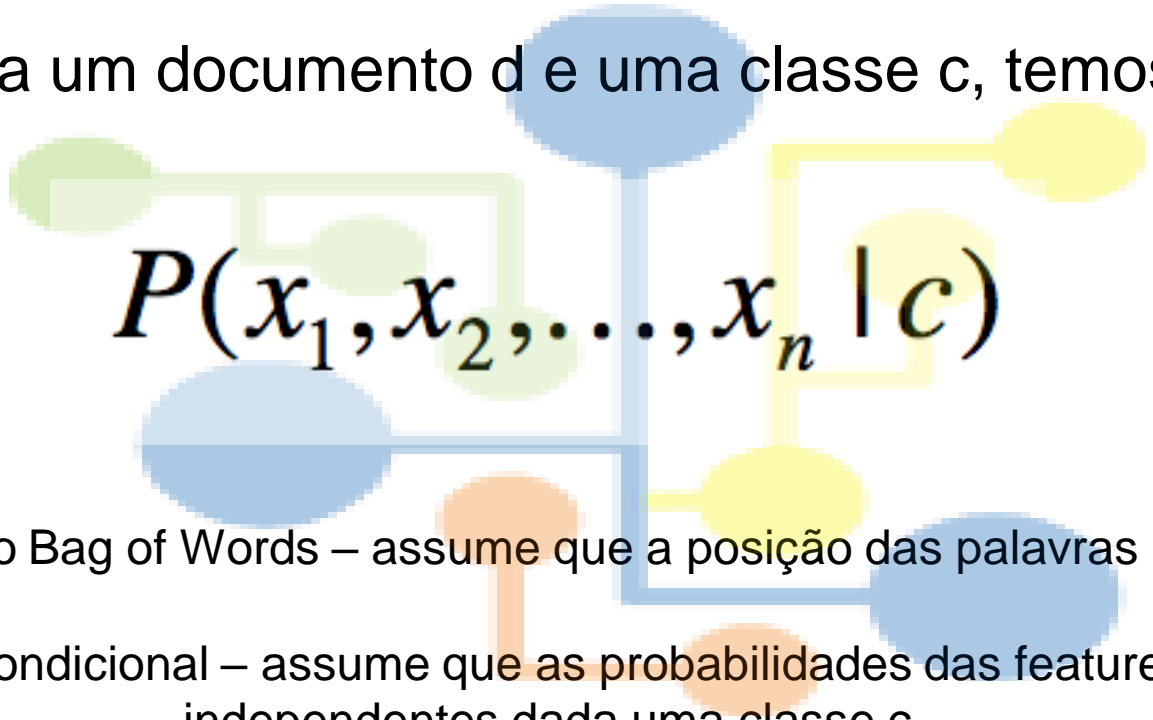
Para calcular $P(c)$ nós contamos a frequência relativa das classes no Corpus

Para calcular $P(x_1, x_2, \dots, x_n)$ precisaríamos de muitos dados, uma vez que teríamos muitos parâmetros



Classificador Naïve Bayes

Para um documento d e uma classe c , temos:

A decorative background diagram consisting of a central blue vertical line. To the left, a green horizontal line connects to a green oval. To the right, a yellow horizontal line connects to a yellow oval. Below the central line, an orange vertical line connects to an orange oval. A blue horizontal line connects the central line to a blue oval on the right. Various other colored ovals and lines are scattered in the background.
$$P(x_1, x_2, \dots, x_n | c)$$

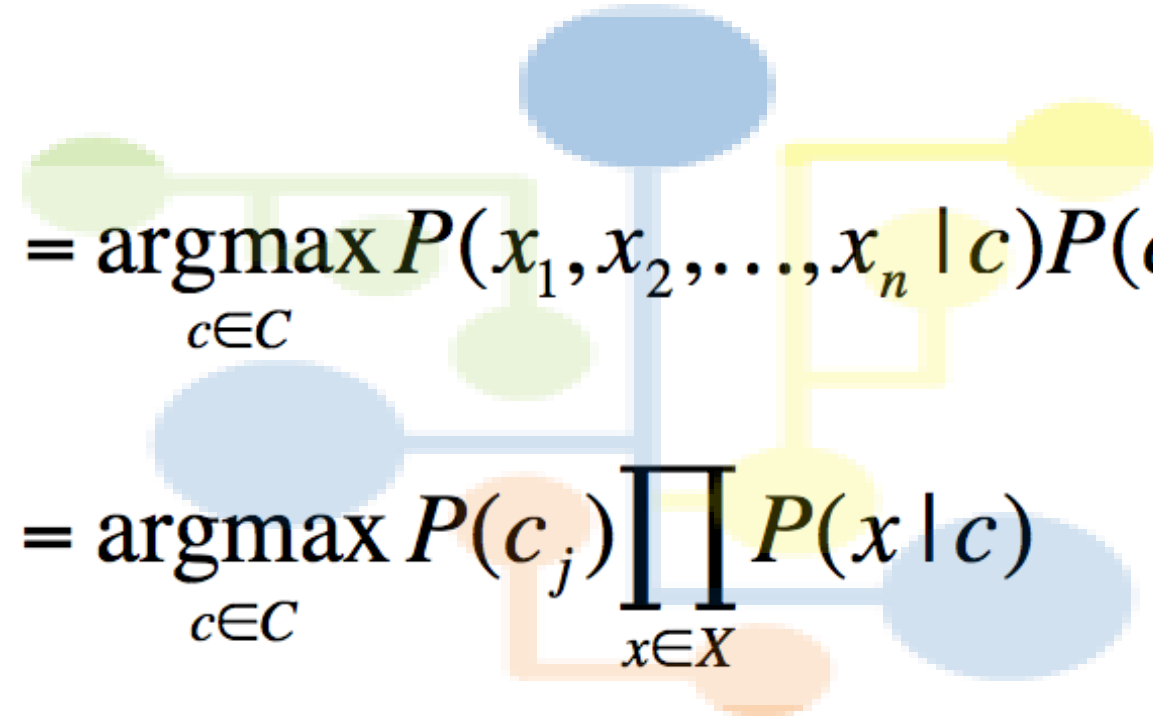
Representação Bag of Words – assume que a posição das palavras não importa

Independência Condicional – assume que as probabilidades das features $P(x_i | c_j)$ são independentes dada uma classe c

$$P(x_1, \dots, x_n | c) = P(x_1 | c) \cdot P(x_2 | c) \cdot P(x_3 | c) \cdot \dots \cdot P(x_n | c)$$



Classificador Naïve Bayes

A faint background diagram of a Naïve Bayes network. It consists of several nodes represented by colored circles (blue, green, yellow, orange) connected by lines. A central blue node is connected to several other nodes, including a green node above it, a yellow node to its right, and an orange node below it. There are also other nodes connected to these, forming a complex web of dependencies.
$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c) P(c)$$

$$c_{NB} = \operatorname{argmax}_{c \in C} P(c_j) \prod_{x \in X} P(x | c)$$



Classificador Naïve Bayes

positions ← Todas as posições de palavras no documento

$$c_{NB} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{i \in \text{positions}} P(x_i | c_j)$$



Aprendizado de Parâmetros no Classificador Naïve Bayes



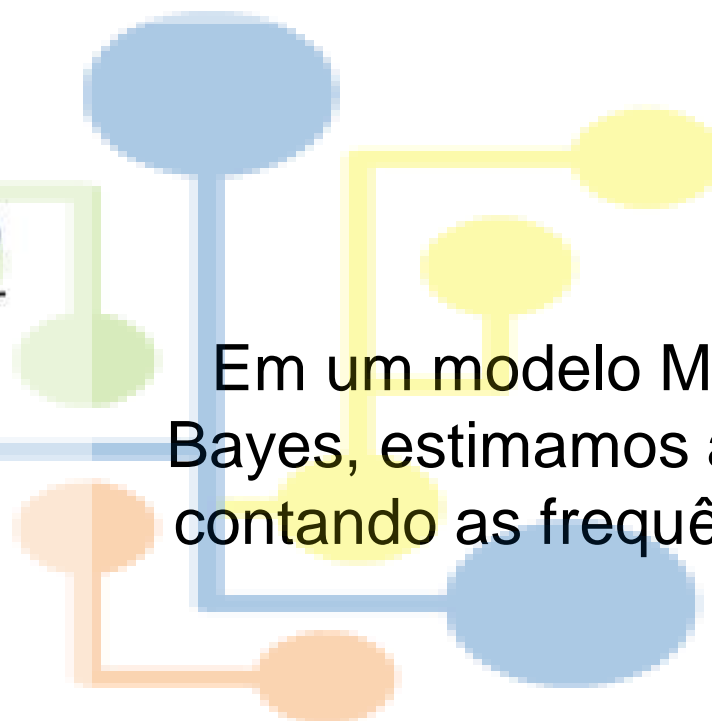


Aprendizado de Parâmetros no Classificador Naïve Bayes

$$\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

Em um modelo Multinomial Naive Bayes, estimamos as probabilidades contando as frequências nos dados.





Aprendizado de Parâmetros no Classificador Naïve Bayes

$$\hat{P}(c_j) = \frac{\text{doccount}(C = c_j)}{N_{doc}}$$

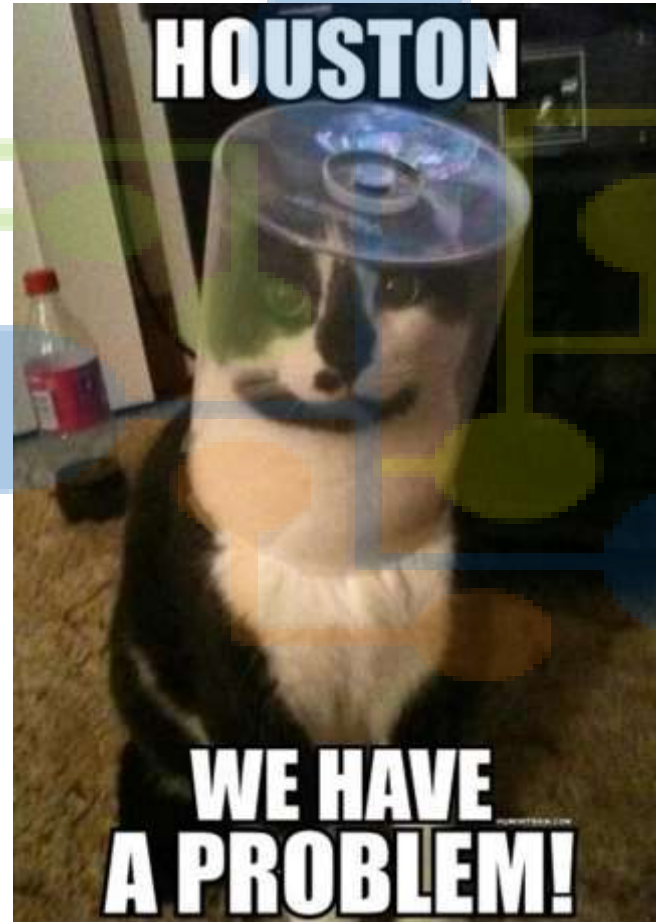
$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

Criamos um mega-documento para a classe j (por exemplo), concatenando todos os documentos com este tópico.

Calculamos a frequência de w neste documento.



Aprendizado de Parâmetros no Classificador Naïve Bayes





Aprendizado de Parâmetros no Classificador Naïve Bayes

Imagine que durante o treinamento, não tenhamos nenhum documento com a palavra *fantastic* e consequentemente não houve qualquer classificação na categoria *positive* (considerando análise de sentimentos)

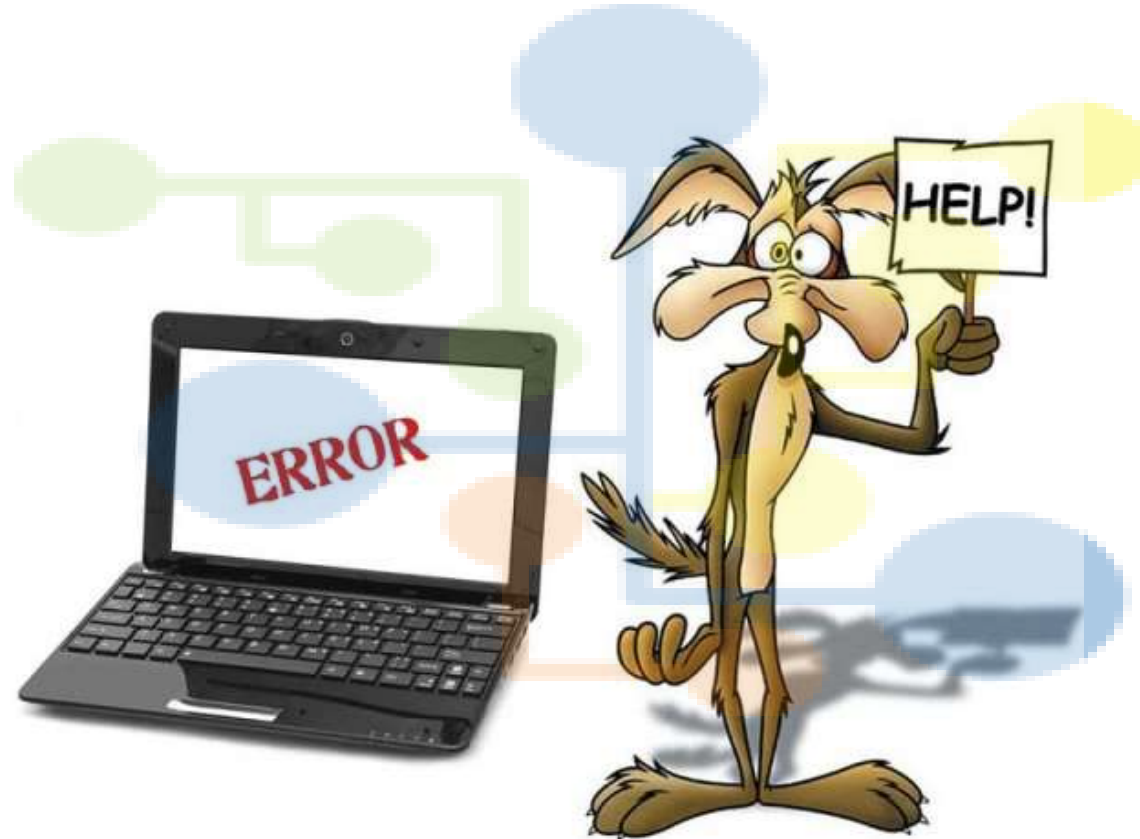
$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{\text{count}(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0$$

Probabilidade zero não pode ser condicionada, não importa qual seja a outra evidência.

$$c_{MAP} = \operatorname{argmax}_c \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$



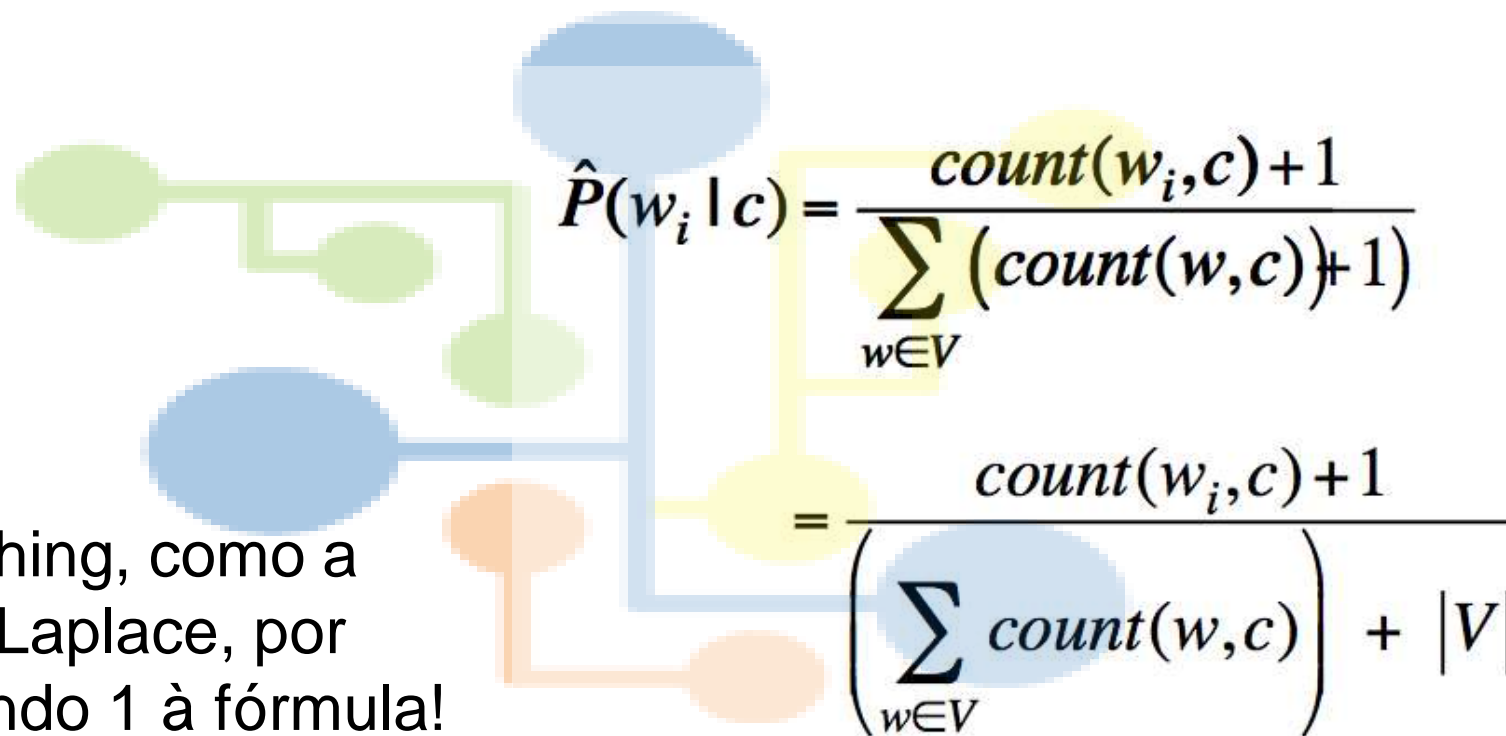
Aprendizado de Parâmetros no Classificador Naïve Bayes





Aprendizado de Parâmetros no Classificador Naïve Bayes

Usamos Smoothing, como a suavização de Laplace, por exemplo, adicionando 1 à fórmula!


$$\hat{P}(w_i | c) = \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)}$$
$$= \frac{\text{count}(w_i, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V|}$$



Data Science
Academy

Data Science Academy felipe.oliveiras2000@gmail.com 5f8a0b3ee32fc37d576ba60d



Obrigado