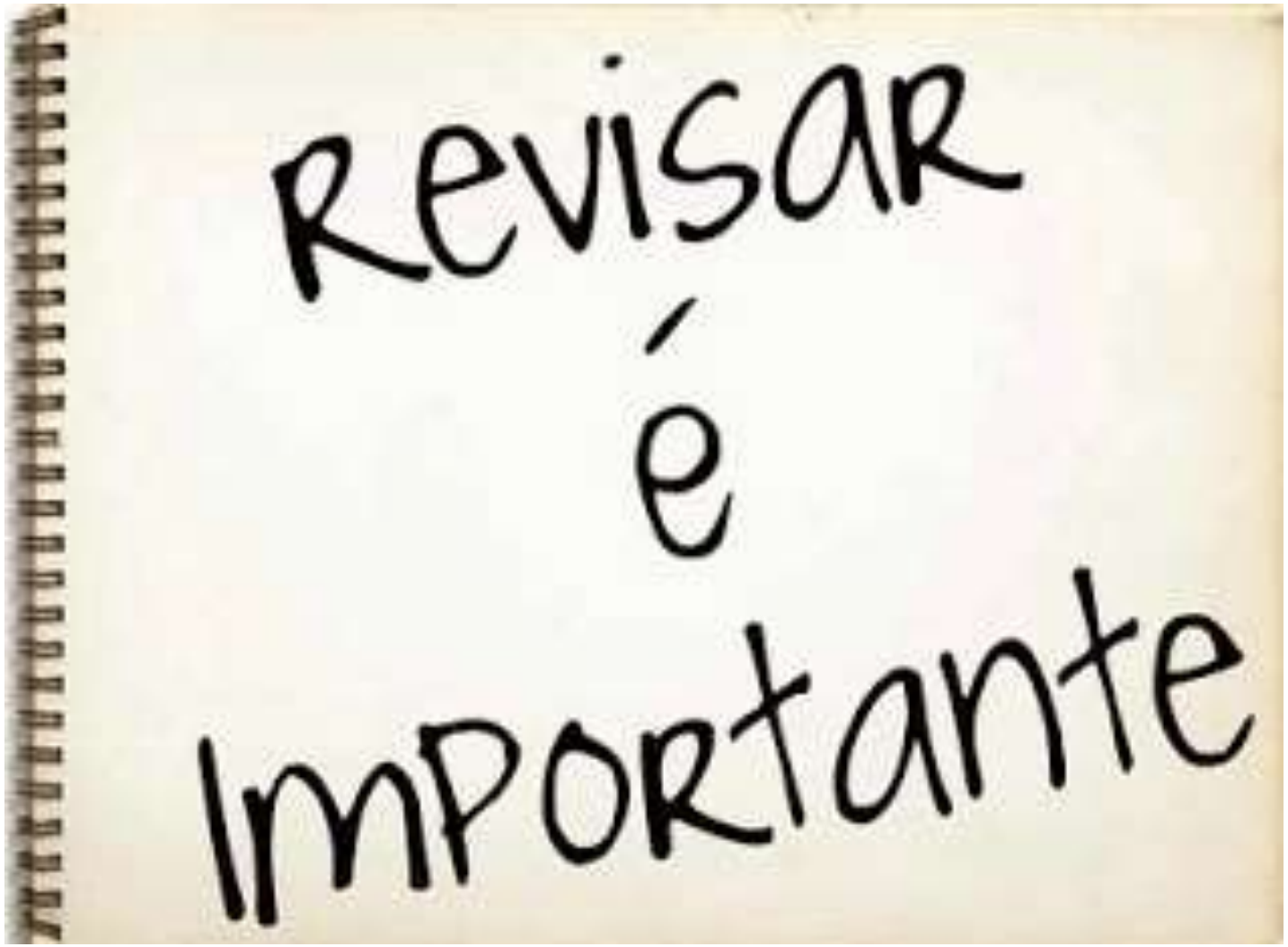
CENTRO UNIVERSITÁRIO
INSTITUTO DE EDUCAÇÃO SUPERIOR DE BRASÍLIA – IESB

Aprendizagem de Máquina

PABLO COELHO FERREIRA

Antes de começarmos...
.



pablo.coelho@gmail.com

Pandas

- É uma biblioteca escrita para programação em Python.
- Serve para analisar, limpar, explorar e manipular dados (Big Data).
- Oferece estrutura e operações para manipular tabelas numéricas e séries temporais.
- É software livre rodando sob a licença Three-Clause BSD.

Etiquetando os dados (nomes para colunas)

```
import pandas as pd

a = [1, 7, 2]

myvar = pd.Series(a, index = ["x", "y", "z"])

print(myvar)

print(myvar[“y”])
```

Saída do código

x 1

y 7

z 2

7

Séries a partir de dicionários

```
import pandas as pd

calories = {"dia 1": 420, "dia 2": 380, "dia 3": 390}

myvar = pd.Series(calories)

print(myvar)
```

Saída do código

dia 1 420

dia 2 380

dia 3 390

DataFrame (tabela completa)

```
import pandas as pd

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pd.DataFrame(mydataset)

print(myvar)
```

Saída do código

	cars	passings
0	BMW	3
1	Volvo	7
2	Ford	2

Estrutura com duas dimensões. Similar a um vetor (array) de duas dimensões ou a uma tabela de dados.

Carregando um arquivo para um DataFrame

- import pandas as pd
- df = pd.read_csv('data.csv')
- print(df)

Saída do código

	nome	dia nascimento	mes nascimento	uf	cidade
0	Pablo	26	1	df	brasilia
1	Enzo	6	5	df	brasilia
2	Tiago	2	3	df	florianopolis
3	maria	26	4	sp	sao paulo

Arquivo dados2.csv

- Contêm 169 registros referentes a Duração, Pulsação, Max e Calorias.

Duracao,Pulso,Max,Calorias
60,110,130,409.1
60,117,145,479.0
60,103,135,340.0
45,109,175,282.4
45,117,148,406.0
60,102,127,300.0
60,110,136,374.0
45,104,134,253.3
(...)

to_string

```
import pandas as pd

df = pd.read_csv('dados2.csv')

print(df.to_string())
```

Utilize o to_string para imprimir todos os dados de um DataFrame. Caso contrário vai imprimir apenas os cinco primeiros e os cinco últimos.

Exemplo – sem o to_string

	Duracao	Pulso	Max	Calorias
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0
..	...	...	...	...
164	60	105	140	290.8
165	60	110	145	300.0
166	60	115	145	310.2
167	75	120	150	320.4
168	75	125	150	330.4

Lendo todos os itens de uma coluna

```
import pandas as pd

df = pd.read_csv(<nome arquivo>)
vetDados = df[<nome coluna>]

for item in vetDados:
    <Código onde ”item” tem os dados da coluna>
```

pablo.coelho@gmail.com

Número máximo de registros retornados

```
import pandas as pd

print(pd.options.display.max_rows)
```

Vai mostrar o número máximos de linhas (registros) que um Dataframe vai retornar. Se existir mais registros do que o max_rows, o Pandas trás o cabeçalho e os cinco primeiros e últimos registros.

Esse valor pode ser alterado: **pd.options.display.max_rows = 9999**

Lendo JSON

- Arquivos JSON são arquivos texto formatados como objetos.

```
import pandas as pd

df = pd.read_json('data.json')

print(df.to_string())
```

Arquivo JSON


```
{
   "Duration":{
     "0":60,
     "1":60,
(...)
 },
   "Pulse":{
     "0":110,
     "1":117,
(...)
 },
   "Maxpulse":{
     "0":130,
     "1":145,
(...)
 }
 }
```

head e tail

```
import pandas as pd

df = pd.read_csv('dados2.csv')

print("cabecalho")
print(df.head())
print("cabecalho (dez primeiros)")
print(df.head(10))
print("cauda")
print(df.tail())
print("cauda (dez últimos)")
print(df.tail(10))
```

Saída do código

cabecalho

	Duracao	Pulso	Max	Calorias
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0

cabecalho (dez primeiros)

	Duracao	Pulso	Max	Calorias
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0
5	60	102	127	300.0
6	60	110	136	374.0
7	45	104	134	253.3
8	30	109	133	195.1
9	60	98	124	269.0

cauda

	Duracao	Pulso	Max	Calorias
164	60	105	140	290.8
165	60	110	145	300.0
166	60	115	145	310.2
167	75	120	150	320.4
168	75	125	150	330.4

cauda (dez últimos)

	Duracao	Pulso	Max	Calorias
159	30	80	120	240.9
160	30	85	120	250.4
161	45	90	130	260.4
162	45	95	130	270.0
163	45	100	140	280.9
164	60	105	140	290.8
165	60	110	145	300.0
166	60	115	145	310.2
167	75	120	150	320.4
168	75	125	150	330.4

pablo.coelho@gmail.com

Informações sobre os dados

```
print(df.info())

 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Duracao   169 non-null    int64
 1   Pulso     169 non-null    int64
 2   Max       169 non-null    int64
 3   Calorias  164 non-null    float64 #(existem 5 registros sem o dado no
                                       arquivo.)
```

Limpando dados

Daqui para frente iremos realizar os exercícios....

pablo.coelho@gmail.com

Limpeza de dados

- Para uma correta aprendizagem a máquina deve ser “educada” com dados o mais limpo possível.

- Evitar:
  - Células vazias
  - Dados em formato errado
  - Dados errados
  - Dupicados

Retirando registros com célula(s) vazia(s)

import pandas as pd

df = pd.read_csv('dados3.csv’) #Arquivo de dados é o **dados3.csv**

print(df.info()) #32 registros

new_df = df.**dropna**() #Retorna um novo DataFrame, não altera o original

print(df.info()) #31 registros

Para alterar o DataFrame original

```
import pandas as pd

df = pd.read_csv(‘dados3.csv')

df.dropna(inplace = True)

print(df.to_string())
```

Inserindo um valor padrão onde está vazio

```
import pandas as pd

df = pd.read_csv('dados3.csv')

df.fillna(130, inplace = True) #Toda célula vazia receberá o valor 130.
```

Alterando vazio para um valor padrão em uma coluna

```
import pandas as pd

df = pd.read_csv('dados3.csv')

df["Calorias"].fillna(130, inplace = True)
```

Substituindo pelo valor médio, mediana e moda.

```
import pandas as pd

df = pd.read_csv('dados3.csv')

x = df["Calorias"].mean()     #median ou mode()[0]
                              #mode retorna uma série, em cada posição,
                              #em ordem, volta o valor do maior para a
                              #menor quantidade presente na série

df["Calorias"].fillna(x, inplace = True)
```

Correção de tipo de valor

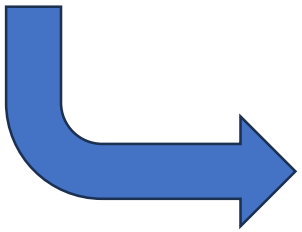
```
import pandas as pd

df = pd.read_csv('dados3.csv')

print(df.to_string())

df['data'] = pd.to_datetime(df['data'], format='mixed') # vai corrigir os
                                                        # registros 22 (NaT) e
                                                        # 26 (vai formatar a data)

print(df.to_string())
```

Limpar linhas que não tenham datas

- Mesmo código anterior, acrescentando a linha:

df.dropna(subset=['data'], inplace = True)

21	60	2020-12-21	108	131	364.2
22	45	NaT	100	119	282.0
23	60	2020-12-23	130	101	300.0



21	60	2020-12-21	108	131	364.2
23	60	2020-12-23	130	101	300.0

Ajustando dados errados

```
import pandas as pd

df = pd.read_csv('dados3.csv')

print(df.to_string())

df.loc[7, 'Duracao'] = 45 # ajusta o registro 7 para o valor desejado.

print(df.to_string())
```

Pode ser utilizado para vários registros

```
import pandas as pd

df = pd.read_csv('dados3.csv')

for x in df.index:
  if df.loc[x, "Duracao"] > 120:
    df.loc[x, "Duracao"] = 120 # Altera o valor da coluna Duracao de
                                  # uma determinada linha
OU
for x in df.index:
  if df.loc[x, "Duracao"] > 120:
    df.drop(x, inplace = True)   # Elimina a linha
```

Removendo duplicatas

```
import pandas as pd

df = pd.read_csv('dados3.csv')

print(df.duplicated()) # Mostra as linhas duplicadas

df.drop_duplicates(inplace = True) # Remove as linhas duplicadas

print(df.to_string())
```

Plotando os dados em um gráfico

```
import pandas as pd

import matplotlib.pyplot as plt

df = pd.read_csv('dados3.csv')

df.plot()

plt.show()
```

Gráfico de Pizza

```
import pandas as pd
from matplotlib import pyplot as plt

df = pd.read_csv('dados4.csv')
from matplotlib import pyplot as plt
import numpy as np

numA = 0
numB = 0
vetNumeros = df['survived']
```

```
for item in vetNumeros:
    if item == 0:  #verifica se o registro
                   #é 0. A coluna só tem
                   #registros 0 e 1.
        numA += 1
    else:
        numB += 1

dados = [numA,numB]
plt.pie(dados, labels = ['Números 0',' Números 1'])
plt.show()
```

Gráfico de dispersão

plt.scatter (x, y)

plt.title(<Título do Gráfico>)

plt.xlabel(<Label do eixo X>)

plt.ylabel(<Label do eixo X>)

Dever de casa

Trabalhando com Sets:

Crie uma lista contendo os seguintes elementos: "maçã", "banana", "laranja", "uva", "maçã", "melão", "mamão" e "banana".

Crie um arquivo chamado minhas_frutas.txt e grave nele os nomes das frutas do conjunto, com uma quantidade aleatória entre 0 e 100 para cada registro.

.

Abra o arquivo, leia e exiba seu conteúdo no console através de um Data Frame com os nomes das colunas: Fruta e quantidade

Atente-se que há frutas repetidas no arquivo e as suas quantidades devem ser somadas.

pablo.coelho@gmail.com

Trabalho

Descrição do Trabalho

- O grupo deverá desenvolver um **analisador de dados** que carregue, processe e visualize informações de um conjunto de dados fornecido*.

- O sistema deve ser implementado em **Python**, utilizando as bibliotecas **Pandas** para manipulação de dados e **Matplotlib** para visualização gráfica. O trabalho deverá seguir o padrão de desenvolvimento **PEP 8 – Style Guide for Python Code** (referência) **(1 ponto).**

*https://www.kaggle.com/datasets/mahmoudelhemaly/students-grading-dataset

Trabalho – Apresentar na aula seguinte a prova.

Requisitos do Projeto – Itens:

1. O programa deve permitir o carregamento de um arquivo **CSV ou JSON** com dados de alunos (students-grading-dataset). (1 ponto)
    1. Perguntar ao usuário o caminho do arquivo
    2. O usuário deve poder visualizar um resumo estatístico dos dados carregados.
        1. Quantidade de dados carregados
        2. Quantidade de homens e mulheres
        3. Quantos registros sem dados sobre a educação dos pais.
2. Limpeza de dados (3 pontos)
    1. Remover os registros que tem a educação dos pais vazios.
    2. Alterar os dados de presença (Attendance) que estão null para a mediana.
    3. Apresentar o somatório de Attendance.
3. Consulta a dados (2 pontos)
    1. O usuário pode escolher uma das colunas e o sistema deve apresentar: média, mediana, moda, desvio padrão dos dados daquela coluna.
4. Gráficos (3 pontos)
    1. O sistema deve produzir gráfico de dispersão para “horas de sono” x “nota final”
    2. Gráfico de barras – idade x média das notas intermediárias (midterm_Score)
    3. Gráfico de pizza para as idades (Agrupadas: até 17; 18 a 21; 21 a 24; 25 ou mais).

pablo.coelho@gmail.com

Critérios

- O sistema que não importar os dados OU travar durante a execução do código, seja por bug ou por não validação de entrada de usuário – 0 PONTOS.
  - Validar, entre outros, se o caminho do arquivo existe, se a coluna para a qual se está pedindo as médias, modas... É uma coluna numérica.

- Pontos extras (o máximo de pontos do trabalho é 10! Se passar, fica com apenas com 10)
  - 1 ponto: Documentação utilizando docstrings.
    - O resultado deve ser apresentado utilizando o Sphinx, pydoc ou MkDocs.
  - 0,1 ponto para cada função que tiver controle de erro (máximo 1 ponto).
  - 1 ponto se o sistema tiver um log de ações do usuário (todas).
    - Nesse caso deve pedir o nome do usuário como entrada (nesse caso não precisa ter crítica, exceto se é um nome composto APENAS por caracteres, com ao menos 3. Os nomes podem ser duplicados).
  - 1 ponto se o git do projeto “contar” a história da evolução. Com commits de todos alunos, evoluções, controle de bugs, etc...

Regras

- Grupos de até quatro integrantes.

- No dia da apresentação todos devem estar presentes. Quem não estiver não recebe os pontos do projeto.
  - Em caso de atestado médico, o aluno poderá fazer a apresentação na aula seguinte e/ou participar por vídeo – responsabilidade do grupo conseguir o acesso.

- A nota é do grupo. Perguntas serão feitas para os membros do grupo durante a apresentação sobre o código apresentado.

- As apresentações podem ser iniciadas até (quantidade de grupos restante)*15minutos – horário de término da aula.
  - Se estourar esse prazo, o grupo da vez será sorteado.
  - Ao iniciar a apresentação não pode alterar mais nada (Arquivos, códigos, parâmetros, caminhos, permissões....)

Aula 05

pablo.coelho@gmail.com

SCIKIT Learn

pablo.coelho@gmail.com

SCIKIT Learn

- É uma biblioteca para aprendizagem de máquina
- Suporta aprendizagem supervisionada e não supervisionada
- Open Source (BSD)

- Provê ferramentas para pré-processamento de dados, seleção, ajuste e avaliação de modelos, além de outras ferramentas para AM.

estimators

- Modelos de aprendizagem e predição.

- Separados em três categorias:
  - Supervisionados
  - Não supervisionados
  - Transformação

Tipos de dados

- Numéricos:
  - Arrays do Numpy
  - DataFrames do Pandas

- Categóricos:
  - Trata dados não numéricos

Como é o funcionamento de uma máquina que aprende?

- Um conjunto de dados com características e resultados é carregado no modelo de aprendizagem.
  - Método **fit**

- Na sequência é feita a predição.
  - Método **predict**

- A seguir um exemplo didático de como pode ser utilizada uma máquina de prever salários.

Caracterísiticas e resultados

- Lista (array) com variáveis X que levam a um resultado Y.

- Exemplo:

(instituição, curso, ano de conclusão, nota média) (salário inicial)

X	y
[‘IESB’,’ADS’,’2021’,’9’]	[’3.500’]
[‘IESB’,’ADS’,’2021’,’7’]	[’2.000’]
[‘outra’,’ADS’,’2021,’9’]	[’2.500’]
[‘IESB’,’ADS’,’2023’,’9’]	[’4.000’]

pablo.coelho@gmail.com

Predição

A predição responde perguntas feitas para a máquina.

No exemplo HIPOTÉTICO, após treinada, a máquina poderia responder mais ou menos questionamentos assim:

“Cara máquina, qual deverá ser o salário de um aluno que se formar em ADS no IESB em 2024 tendo as seguintes notas médias:

9? → R$ 4.200

7? → R$ 2.500

Mas como treinar uma máquina para isso?

- O aprendizado da máquina se dá através de séries de números analisados estatisticamente.
- Então os dados em texto precisam ser classificados, ou seja, categorizados.
- Só assim eles podem gerar registros aprendíveis.
- A seguir um exemplo simplificado e introdutório de uma predição real a partir de números.

Criando uma estimativa simples

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(random_state=0)
X = [[ 1,  2,  3], [11, 12, 13]] # Características
y = [0, 1]  # Resultados
clf.fit(X, y) #Treinamento do modelo
RandomForestClassifier(random_state=0)

print(clf.predict ([[4, 5, 6], [14, 15, 16]]))
```

Esmiuçando

X = [[1, 2, 3], [11, 12, 13]] # **Características**
y = [0, 1] # **Resultados**

Amostra 1 tem as características [1,2,3] e leva ao resultado [0]
Amostra 2 tem as características [11,12,13] e leva ao resultado [1]

Quando provocado sobre amostras que não existem, como:
[4, 5, 6]
[14, 15, 16]

Qual será o resultado?

Resultado do código

[0,1]

O que representam os valores?

- No exemplo são apenas números mesmo.

- MAS... Extrapolando para o mundo real, com milhares de dados, com muito mais características (colunas) e amostras (linhas). Todas numericamente identificadas e aplicadas aos algoritmos estatísticos. O resultado é um conjunto de dados também.

Máquinas que aprendem

- Modelos de Machine Learning falam a língua da matemática: números!

- Todo modelo é, “por baixo dos panos”, um modelo matemático.

- Isso se aplica aos modelos estatísticos clássicos. Se aplica aos modelos de Visão Computacional, que transformam pixels em números. E precisa se aplicar também a modelos de linguagem. Mas como transformamos texto em números? Bom, com Token e Embedding.

Tokens

- Nós precisamos, de alguma forma, encontrar uma via de mão dupla para transformar texto em números e números em texto.
- Chamamos de “Tokenização” o processo de quebrar longas quantidades de texto em unidades menores. Unidades estas que podem ser mapeadas para se tornarem números.
- Todo o processo da criação dos Tokens trabalha então com esse mapa.
- Chamamos esse mapa de vocabulário, ou de *Lexicon* (“léxico”). Para simplificar, vamos enxergar esse mapa como um dicionário. Dicionário esse onde cada token tem uma posição específica. Ou seja, cada token tem um índice.

Tratamento de dados Categóricos

- Para criar um token podemos trabalhar de diversas formas. Uma delas e transformar dados de uma categoria em token.

- Os dados categóricos (strings e outros) precisam ser transformados para dados numéricos na maioria dos modelos de AM.

- Correspondência mais simples é a 1 x 1.
  - Também conhecida como label ou integer encoding
  - Quando é possível trocar um dado por um número específico.
  - Exemplo - cor da pele: Verde, Vermelho, Azul, Verde... Pode ser transformado para 0,1,2,3,4...

Label Enconding

ID	Cor	Forma
1	Verde	Quadrado
2	Vermelho	Triangulo
3	Vermelho	Quadrado
4	Azul	Triangulo
5	Verde	Circulo

ID	Cor	Forma
1	1	1
2	2	2
3	2	1
4	3	2
5	1	3

pablo.coelho@gmail.com

Criando uma coluna de categoria

```
import pandas as pd

setCores = ('verde','vermelho','azul')
dfCores = pd.DataFrame(setCores, columns=['Cores'])

#Convertendo a coluna para o tipo category
dfCores['Cores'] = dfCores['Cores'].astype('category')

#Cria a coluna de categoria
dfCores['Cores_cat'] = dfCores['Cores'].cat.codes
```

Resultado do Código

	Cores	Cores_cat
0	verde	1
1	vermelho	2
2	azul	0

Fazendo a mesma coisa com SCIKIT

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder #Biblioteca do SCIKIT

#Cores Repetidas
setCores = ('verde','vermelho','azul','azul','azul','rosa','vermelho')
dfCores = pd.DataFrame(setCores, columns=['Cores'])

#Criando uma instância do labelencoder
meuLabelEncoder = LabelEncoder()
dfCores['Cores_cat'] = meuLabelEncoder.fit_transform(dfCores['Cores'])
```

Problema do Label Encoding

- Ao transformar uma categoria em um valor numérico o algoritmo de aprendizagem pode criar alguma relação entre os números como, por exemplo, precedência/prioridade.
  - No exemplo anterior o verde ter prioridade sobre o vermelho.

ID	Cor	Forma
1	Verde	Quadrado
2	Vermelho	Triangulo
3	Vermelho	Quadrado
4	Azul	Triangulo
5	Verde	Circulo

ID	Cor	Forma
1	1	1
2	2	2
3	2	1
4	3	2
5	1	3

pablo.coelho@gmail.com

One Hot Encoding

- Para eliminar os problemas de correlação dos dados na transformação de categorias/strings para números, utiliza-se o One-Hot enconding.

- Ao invés de conversão sequencial das categorias, esta técnica cria uma série de colunas para a conversão colocando números 0 ou 1 dependendo da existência ou não da categoria na amostra.

One Hot Encoder

ID	Cor	Forma
1	Verde	Quadrado
2	Vermelho	Triangulo
3	Vermelho	Quadrado
4	Azul	Triangulo
5	Verde	Circulo

ID	Cor	Forma	Azul	Verde	Verm.	Círc.	Quad.	Triang.
1	Verde	Quadrado	0	1	0	0	1	0
2	Vermelho	Triangulo	0	0	1	0	0	1
3	Vermelho	Quadrado	0	0	1	0	1	0
4	Azul	Triangulo	1	0	0	0	0	1
5	Verde	Circulo	0	1	0	1	0	0

pablo.coelho@gmail.com

Código One Hot Encoder

(Incluir na importação)

```
from sklearn.preprocessing import OneHotEncoder
```

dfCores['Cores_cat'] = meuLabelEncoder.fit_transform(dfCores['Cores'])

(abaixo dessa linha acrescentar)

```
enc = OneHotEncoder(handle_unknown='ignore')
dfEnc = pd.DataFrame(enc.fit_transform(dfCores[['Cores_cat']]).toarray())
dfCores = dfCores.join(dfEnc)
```

Resultado do código

	Cores	Cores_cat	0	1	2	3
0	verde	2	0.0	0.0	1.0	0.0
1	vermelho	3	0.0	0.0	0.0	1.0
2	azul	0	1.0	0.0	0.0	0.0
3	azul	0	1.0	0.0	0.0	0.0
4	azul	0	1.0	0.0	0.0	0.0
5	rosa	1	0.0	1.0	0.0	0.0
6	vermelho	3	0.0	0.0	0.0	1.0

One_Hot Encoder

- Depois de criadas as colunas, você deve apagar a coluna com o ordinal da categoria.

dfCores.drop(**['Cores_cat'],** axis=1, inplace=True)

Ou

dfCores.drop(**dfCores.iloc[:,1:2],** axis=1, inplace=True)

Resultado do código

	Cores	0	1	2	3
0	verde	0.0	0.0	1.0	0.0
1	vermelho	0.0	0.0	0.0	1.0
2	azul	1.0	0.0	0.0	0.0
3	azul	1.0	0.0	0.0	0.0
4	azul	1.0	0.0	0.0	0.0
5	rosa	0.0	1.0	0.0	0.0
6	vermelho	0.0	0.0	0.0	1.0

Tokens – por letras

Exemplo de tokens por letras:

LETRA	Indice	LETRA	Indice	LETRA	Indice
A	1	I	9	Q	17
B	2	J	10	R	18
C	3	K	11	S	19
D	4	L	12	T	20
E	5	M	13	U	21
F	6	N	14	V	22
G	7	O	15	X	23
H	8	P	16	Z	24

Vantagem: dicionário simplificado, poucas posições.

Desvantagem: processamento super complexo.

A frase: O IESB tem o melhor curso de ADS.

Receberia o seguinte token:

[15,0,9,5,19,2,0,20,5,13,0,15,0,13,5,12,8,15,18,
0,3,21,18,19,15,0,4,5,0,1,4,19]

pablo.coelho@gmail.com

Tokens – Palavras

Vantagem: Processamento menos complexo.

Desvantagem: Dicionário complexo.

A frase: O IESB tem o melhor curso de ADS.

Receberia o seguinte token:

[1,290,8900,1,98001,8989023,22,564800]

*os números do token são ilustrativos.

- Um vocabulário em língua portuguesa disponível no GitHub, sob licença permissiva, num repositório do Python.pro.br.
- O dicionário tem um pouco mais de 320 mil palavras.
- Agora imagina a complexidade de se treinar um modelo multi-idiomas.
- E também nas palavras que são “criadas” a todo tempo.
- O nosso modelo se depararia com muitos, muitos tokens desconhecidos.

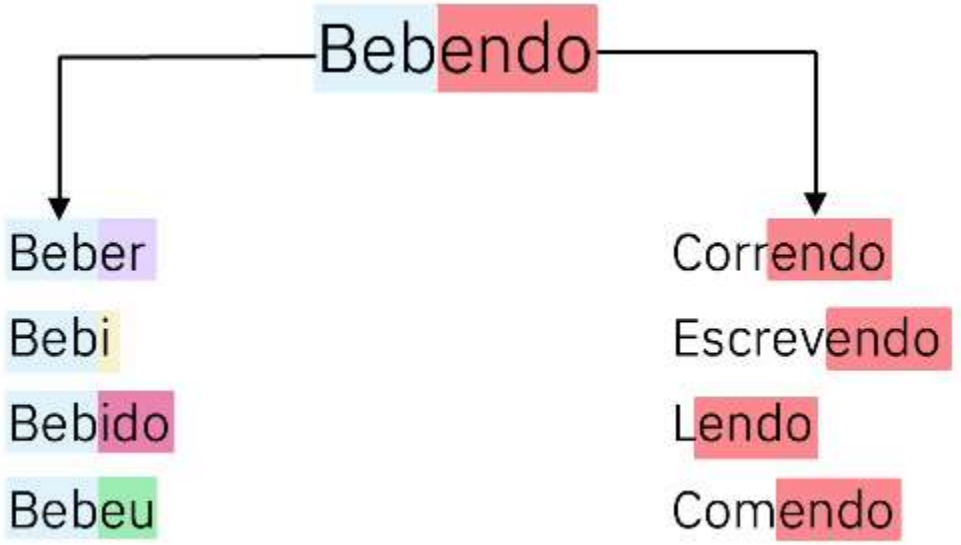
Aumentando a eficiência

- Uma forma mais interessante de lidar com os tokens, é dividindo as palavras. Isolando o radical, prefixo e sufixo. E tratando cada parte como uma unidade diferente, um token diferente.

- Vocês devem ter percebido, que muitas palavras do nosso idioma são formadas com o mesmo radical. Vamos pegar como exemplo a palavra "beber". O início da palavra, o "beb-" pode ser reutilizado em "bebendo", "bebi", "bebido", "bebeu", e assim por diante.

- Também utilizado para palavras compostas.

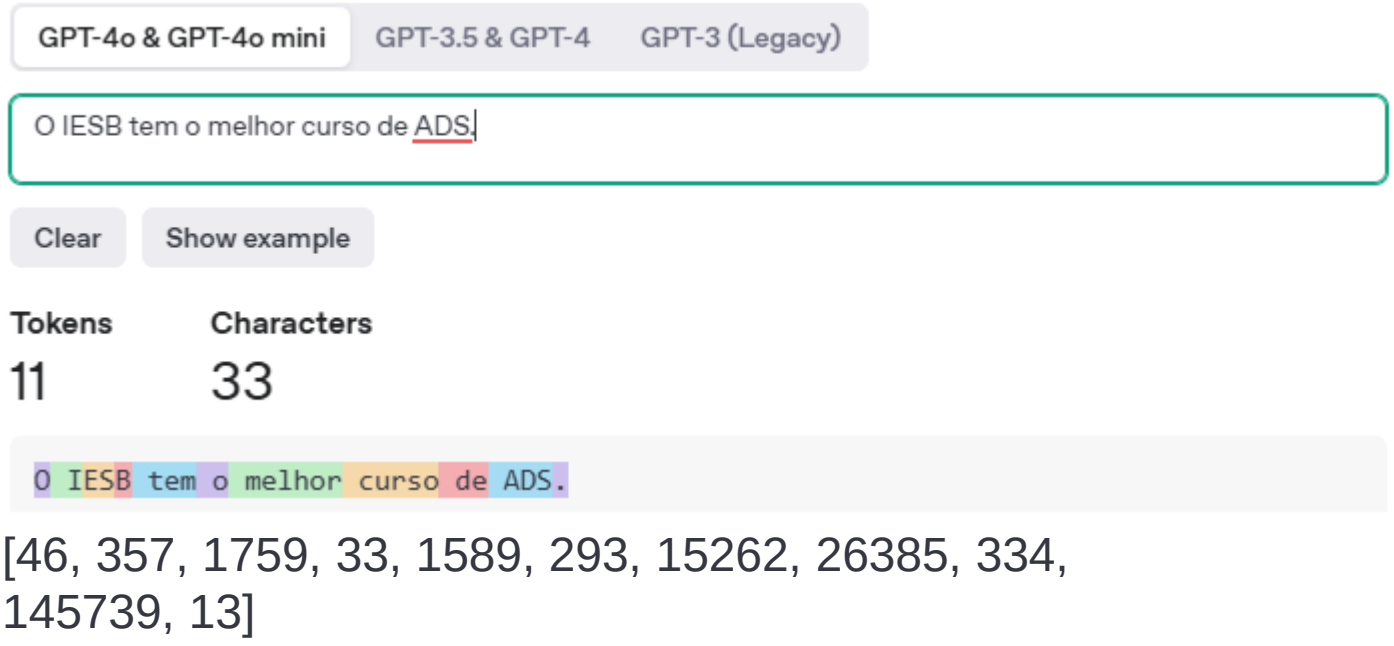
Token por radical, sufixo e prefixo - NGRAMAS



- A resolução de palavras e/ou nomes próprios compostos tem melhor eficiência com esse método.

Tokeninzado pelo ChatGPT

- https://platform.openai.com/tokenizer



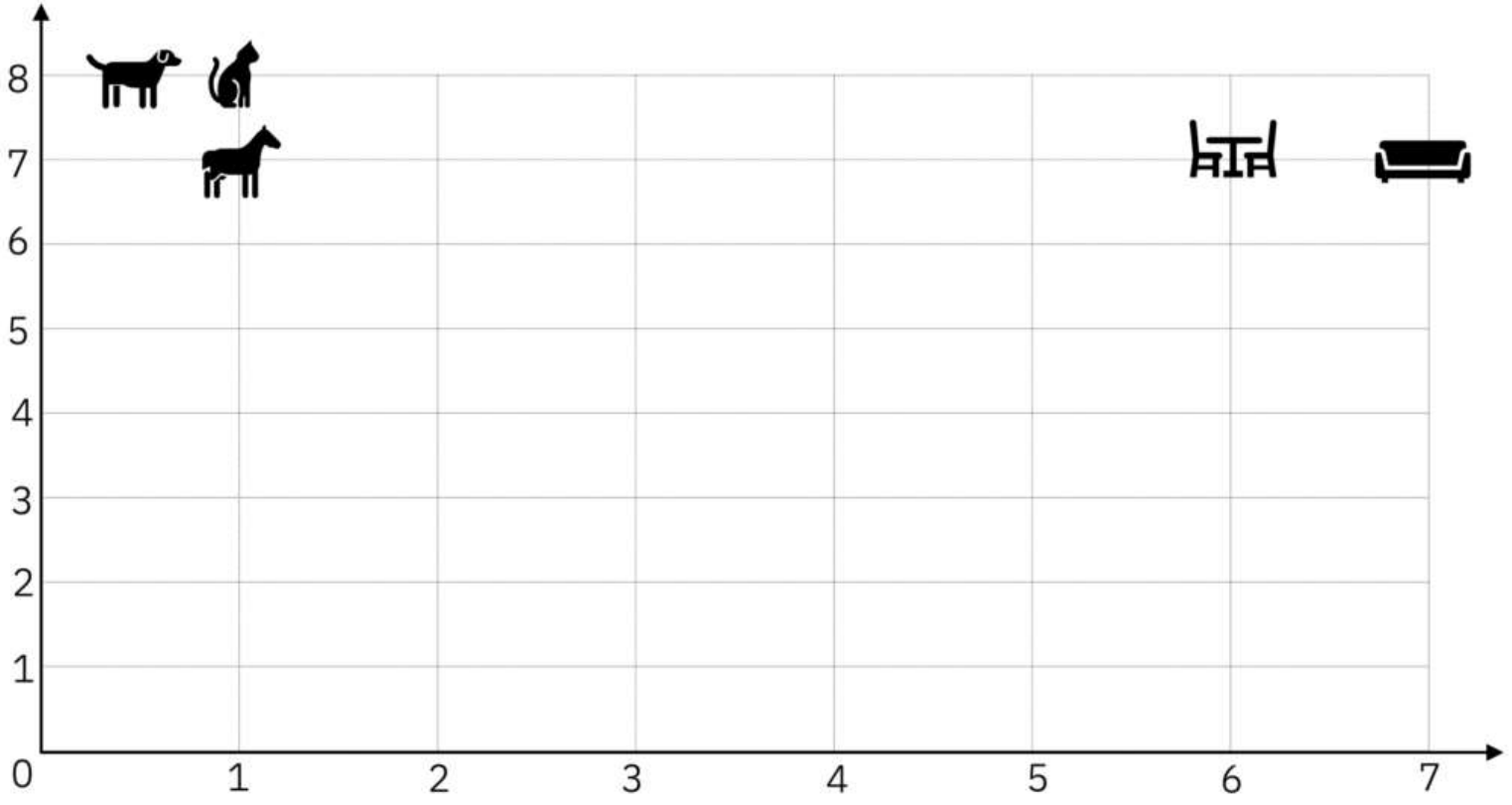
[46, 357, 1759, 33, 1589, 293, 15262, 26385, 334, 145739, 13]

Embedding

- Enquanto o Token é uma representação mais estática de unidades de texto, o Embedding é um pouco mais dinâmico. **Embedding é a representação vetorial do texto em um espaço multidimensional**.

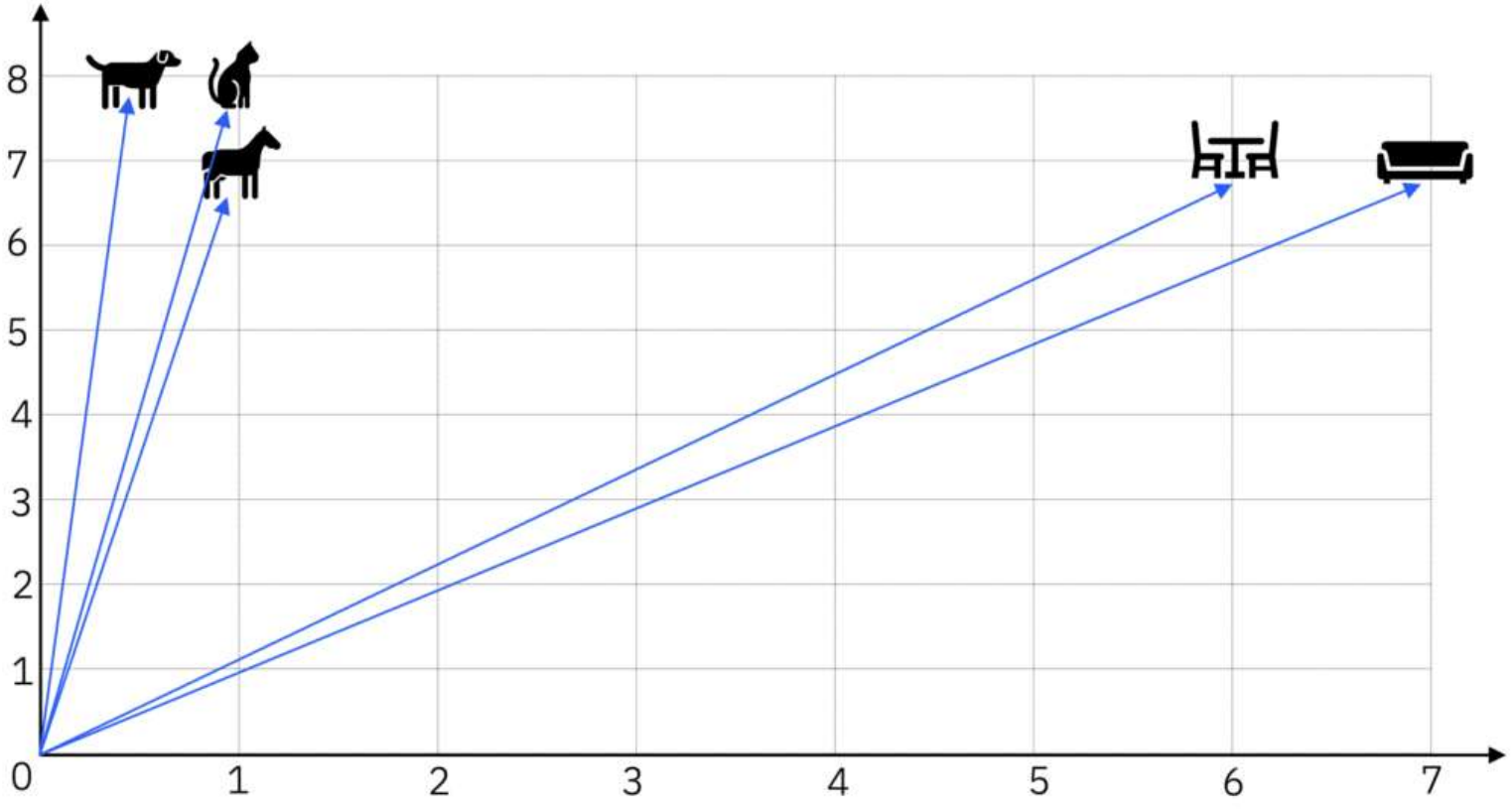
- Nós podemos imaginar os Embeddings como pontos, com coordenadas. Enquanto o Token “IESB" vai ter sempre o ID 47391, o Embedding pode mudar sua posição, dependendo do contexto.

Posição por similaridade



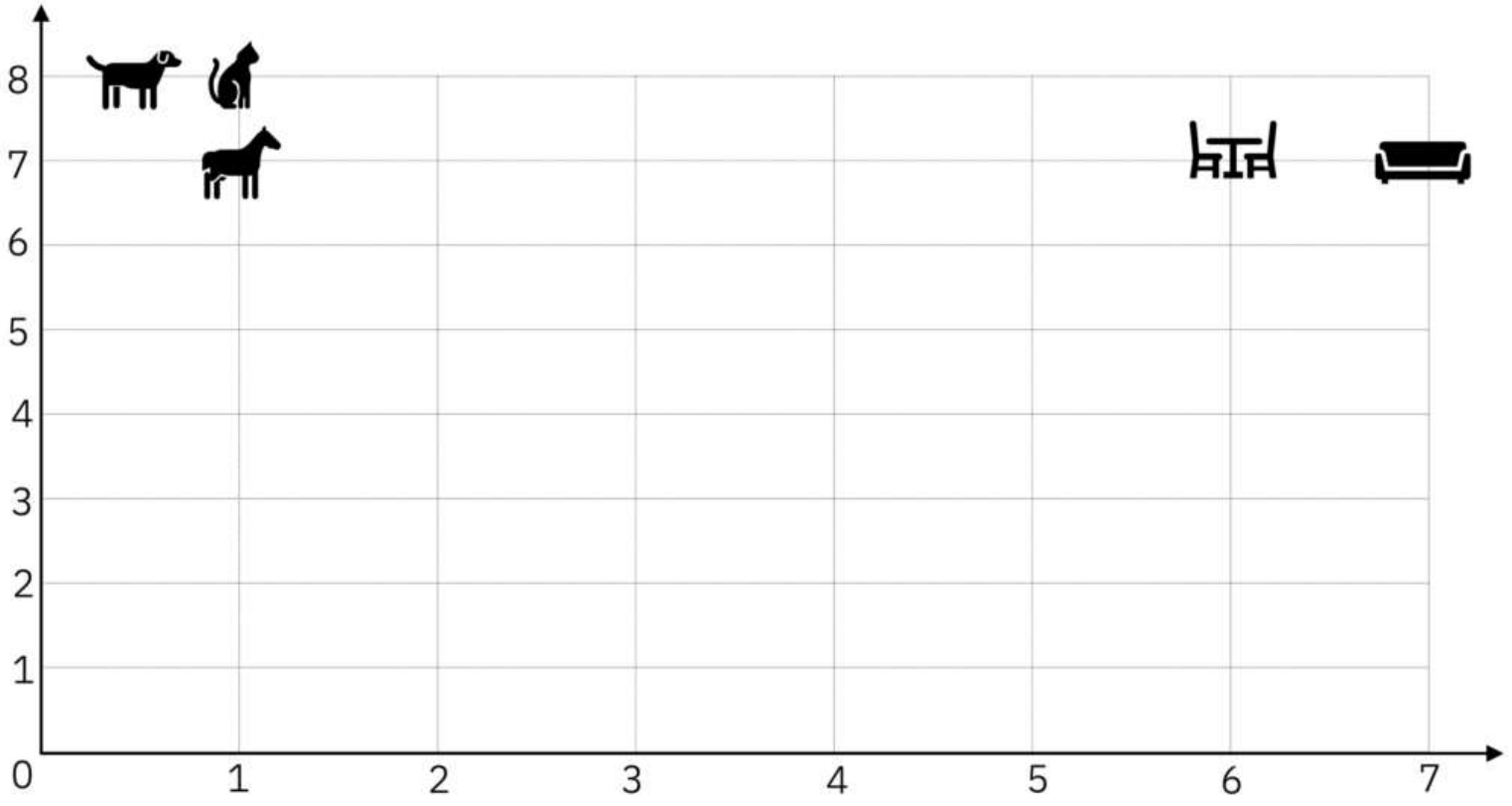
pablo.coelho@gmail.com

Posição por similaridade



pablo.coelho@gmail.com

Posição por similaridade



Elemento – Posição no espaço

(0.5,8)

(1,8)

(1,7)

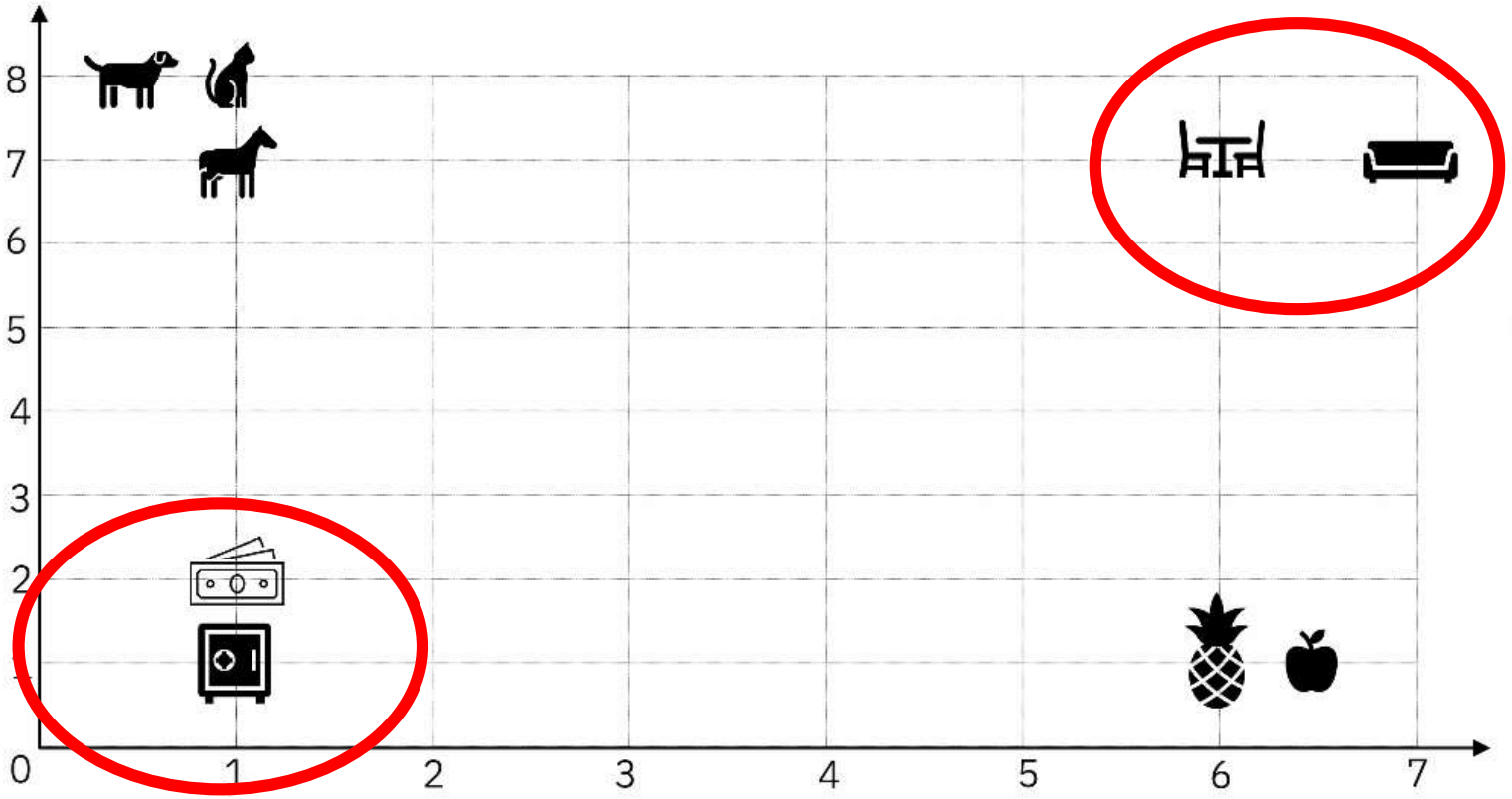
(6,7)

(7,7)

pablo.coelho@gmail.com

Embedding e contexto

A palavra BANCO entraria mais próximo da posição (1,1) ou (6,7)?



pablo.coelho@gmail.com

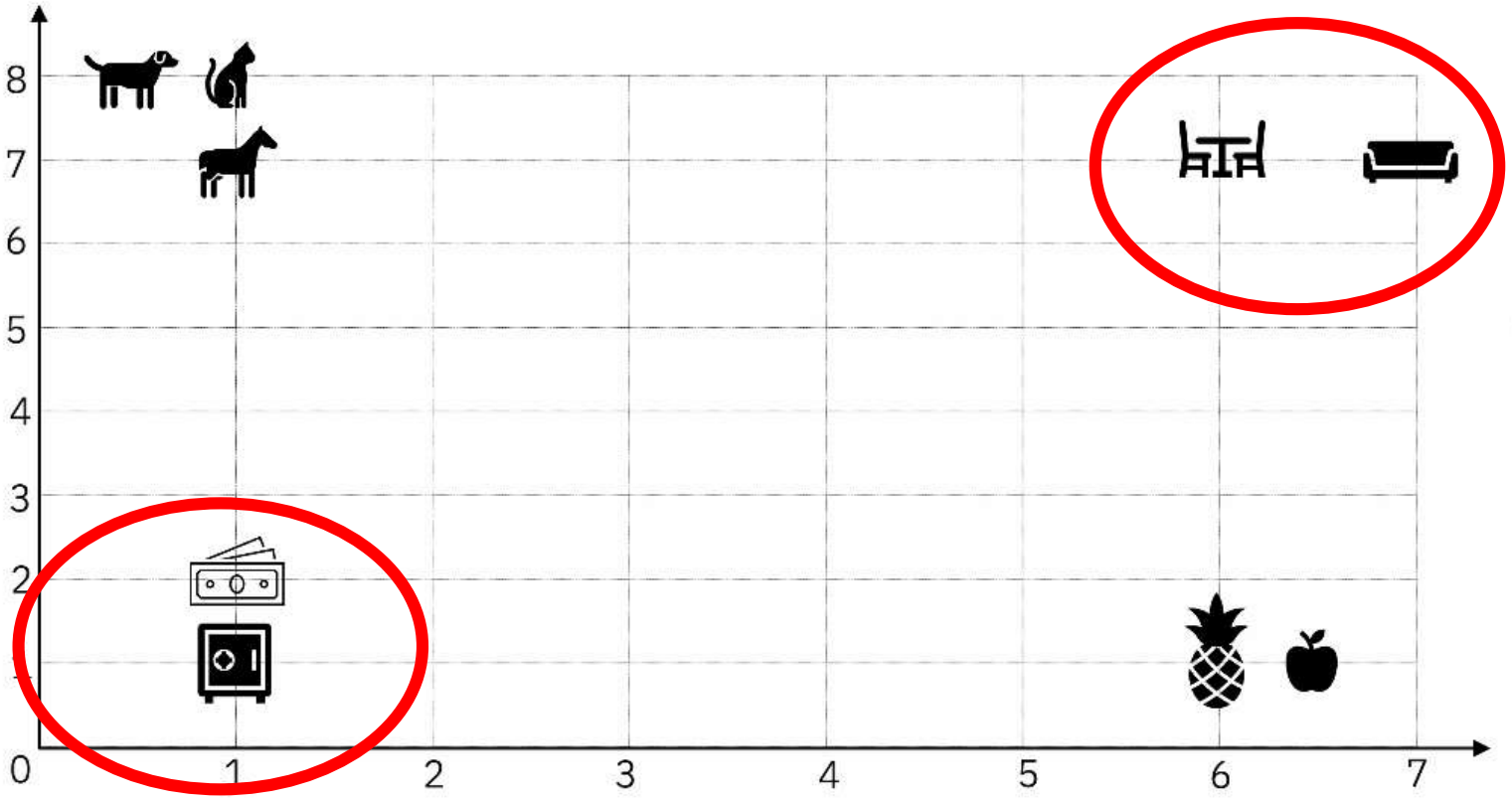
Embedding e contexto

A palavra BANCO entraria mais próximo da posição (1,1) ou (6,7)?

Depende do contexto!

“Vou sentar em um banco.”

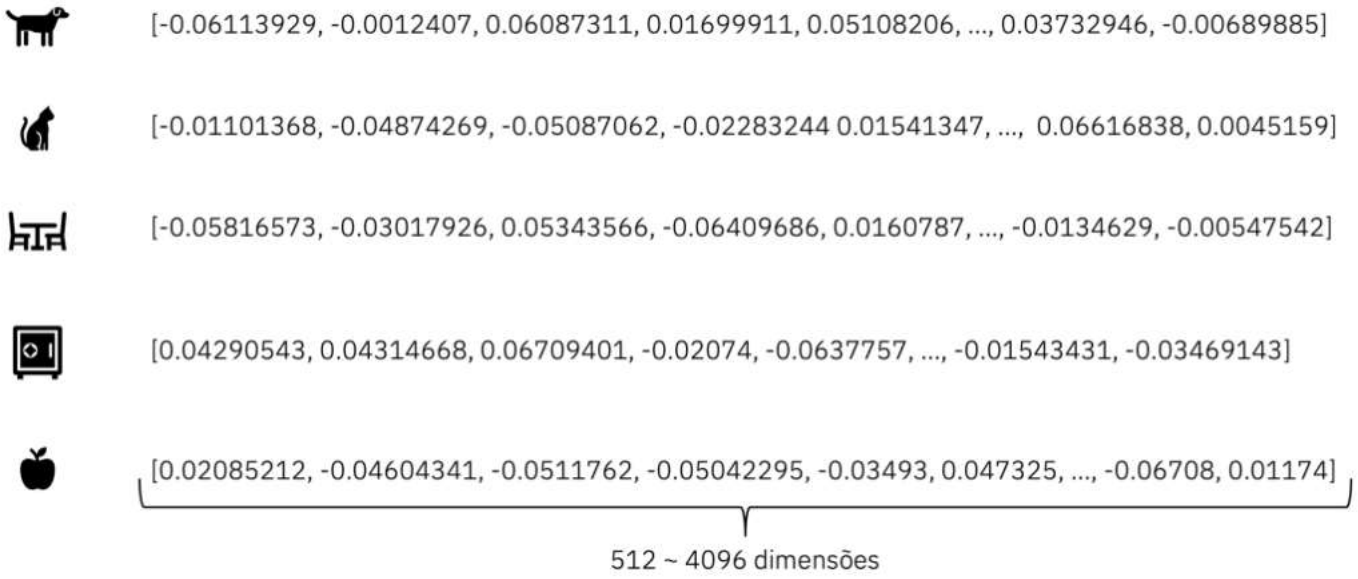
“Vou ao banco fazer um depósito.”



pablo.coelho@gmail.com

Contexto, token e embedding

- Na realidade o registro de uma palavra, considerando suas N possibilidades de aplicação, gera tokens com as seguintes características:



Momento concurso

Um problema comum no processamento de texto é o tratamento de termos compostos por mais de um token, tais como “Ministério Público”, tal que represente uma unidade linguística distinta, em particular na construção de modelos de linguagem.

Considerando o problema acima descrito, a alternativa que apresenta uma técnica usada para sua resolução é:

A) representação por entidade;

B) índice invertido;

C) embedding;

D) representação por n-gramas;

E) decomposição morfológica.

Ano: 2022 **Banca:** FGV **Órgão:** MPE-SC **Prova:** FGV - 2022 - MPE-SC - Analista de Dados e Pesquisa

Dever de casa

Trabalhando com Sets:

Crie uma lista contendo os seguintes elementos: "maçã", "banana", "laranja", "uva", "maçã", "melão", "mamão" e "banana".

Crie um arquivo chamado minhas_frutas.txt e grave nele os nomes das frutas do conjunto, com uma quantidade aleatória entre 0 e 100 para cada registro.

.

Abra o arquivo, leia e exiba seu conteúdo no console através de um Data Frame com os nomes das colunas: Fruta e quantidade

Atente-se que há frutas repetidas no arquivo e as suas quantidades devem ser somadas.

pablo.coelho@gmail.com

Outras fontes de $$$

Editais de subvenção.

https://www.fap.df.gov.br/

pablo.coelho@gmail.com

O edital FAPDF Participa integra o Programa de Difusão Científica da Fundação. O objetivo é incentivar a participação de pesquisadores, profissionais e estudantes em eventos e cursos de curta duração, além de visitas técnicas de caráter científico, tecnológico e inovador. O Participa oferece apoio financeiro para atividades no Brasil ou no exterior. Os candidatos precisam ter vínculo com Instituições de Ensino Superior ou de pesquisa, públicas ou privadas do DF.

A iniciativa busca fortalecer a internacionalização e a troca de conhecimento, ampliando as oportunidades de desenvolvimento científico e tecnológico para os participantes e suas instituições.

O edital FAPDF Publica também faz parte do Programa de Difusão Científica da Fundação, com foco no apoio financeiro para publicação de artigos científicos em revistas nacionais e internacionais de alta especialização. O objetivo é incentivar a difusão do conhecimento gerado por pesquisadores vinculados a instituições de ensino, pesquisa, empresas tecnológicas e organizações da sociedade civil do Distrito Federal.

O edital é voltado para pesquisadores que possuam vínculo com instituições brasileiras sediadas no DF, incluindo startups e organizações que já tenham recebido apoio da FAPDF. A submissão das propostas segue os critérios estabelecidos no edital, respeitando a legislação aplicável.

pablo.coelho@gmail.com

Dever de casa

- Crie um arquivo CSV com 10 entradas conhecidas indicando:
  - IMC, Obeso (true/false)
  - Treine uma máquina
  - Pergunte a ela se um dado IMC (fora do range das entradas) é obeso ou não.

Links úteis

- Comunidade de AM e IA para quem quiser se aprofundar no assunto:
  - https://huggingface.co/
  - https://brains.dev/2022/brains-brazilian-ai-networks/

- Tokens
  - https://www.youtube.com/watch?v=Am73u_4y0ok
- Embeddings
  - https://www.youtube.com/watch?v=_G_--YC5Xd4
  - https://www.youtube.com/watch?v=ygRurMXa5uw


Thats all Folks!