

# **Aula 5 - Geoprocessamento e ML**

Introdução ao aprendizado de máquina - UEMA 2025

---

Thiago S. F. Silva

2025-12-17

## **Parte I - Introdução e Terminologia**

---

# Sensoriamento Remoto

O que é sensoriamento remoto?

# Sensoriamento Remoto

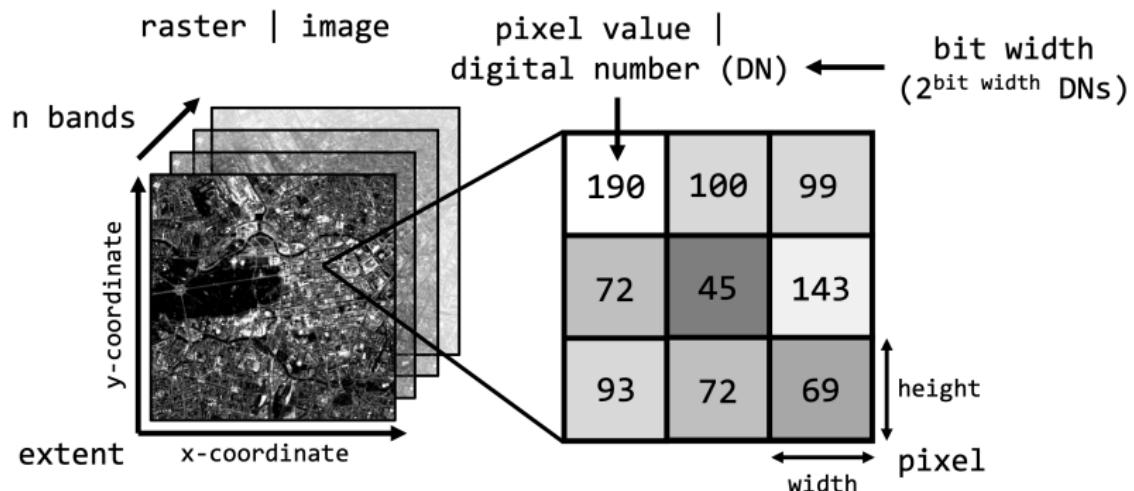
O que é sensoriamento remoto?

- Uma *medida* da quantidade de REM refletida pela superfície.
- Essa medida é organizada na forma de uma *imagem*.

Geoprocessamento: SR + SIG + GNSS

# Estrutura de uma imagem de SR

- Formato **raster**: bandas, pixels, ND (ou DN)



# Dados vetoriais

O famoso 'shapefile' (geopackage é um formato muito melhor).

## POINTS: Individual $x, y$ locations.

ex: Center point of plot locations, tower locations, sampling locations.



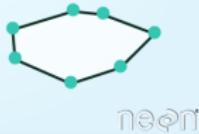
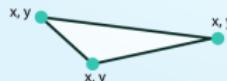
## LINES: Composed of many (at least 2) vertices, or points, that are connected.

ex: Roads and streams.



## POLYGONS: 3 or more vertices that are connected and closed.

ex: Building boundaries and lakes.



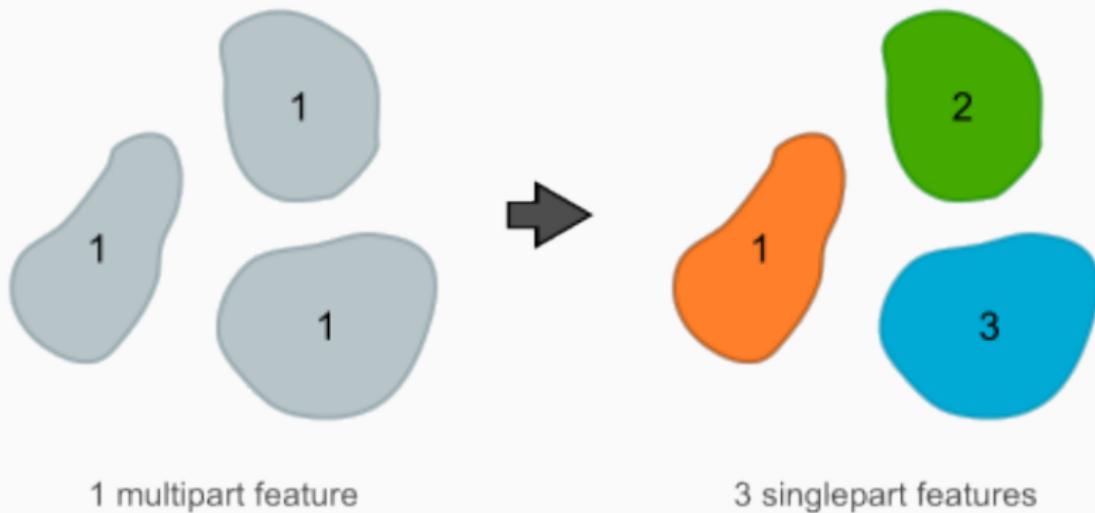
NEON

## Dados vetoriais

- Cada geometria de objeto é conectada a uma *tabela de atributos*.
- Uma tabela de atributos é uma tabela...e já sabemos fazer muita coisa com tabelas!

# Casos problemáticos

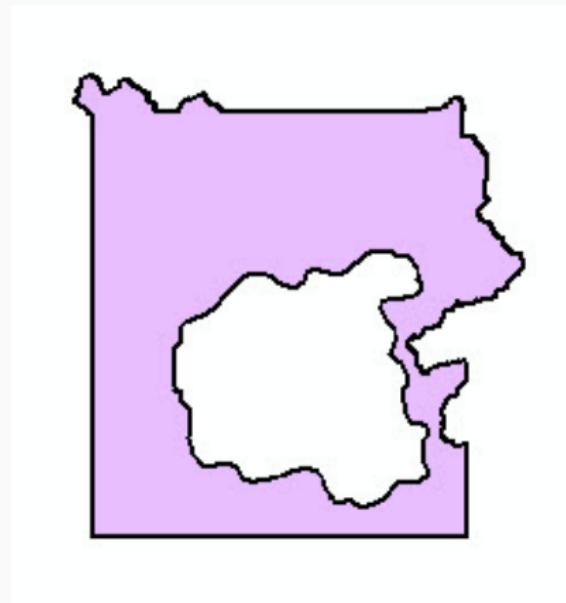
- Polígono Multiparte



<https://grass.osgeo.org/grass-stable/manuals/addons/v.multi2singlepart.html>

## Casos problemáticos

- Polígono em anel (ou 'donut')



<https://esri.github.io/geometry-api-java/doc/Polygon.html>

## Dados geoespaciais - sistema de coordenadas

- O que torna um dado espacial: *georeferenciamento*.
- Exige um **Sistema de Coordenadas de Referência** (Coordinate Reference System, CRS)
- A famosa “projecão”...

# Dados geoespaciais - sistema de coordenadas

- O que torna um dado espacial: *georeferenciamento*.
- Exige um **Sistema de Coordenadas de Referência** (Coordinate Reference System, CRS)
- A famosa “projecão”...
- Um CRS inclui:
  - Datum
  - Projeção
  - Sistema de Coordenadas
  - Origem

## O grande poster explicativo

- Preparado para minha disciplina de GIS
- Disponível na página do curso

# Pacotes para geoprocessamento R

A nova geração:

- `terra` ([link](#)): para rasters (mas também trabalha com vetores)
- `sf` ([link](#)): para vetores

Old school:

- `raster` ([link](#)): adivinhem pra que?
- `sp` ([link](#)): para vetores (mas também suporta rasters)

# Visualizando dados geoespaciais

- `mapview` : <https://r-spatial.github.io/mapview/>
- `tmap` : <https://r-tmap.github.io/tmap/>
- `ggmap` : <https://paleolimbot.github.io/ggspatial/>

# Interface com a nuvem

Muito do sensoriamento remoto moderno envolve operações na nuvem ou em contato com a nuvem. Exemplos:

Google Earth Engine: `rgee`<sup>1</sup>

STAC catalogs: `rstac`

Menção honrosa: `rsi`

---

<sup>1</sup>usa o python por trás das cortinas.

## Usando o `rstac` para acesar e baixar dados

Exemplo: [Brazil Data Cube](#)

[Coleção CBERS4-MUX-2M-1](#): imagens compostas do sensor CBERS-4 MUX (20m) para períodos bimensais, usando o algoritmo CFMASK.

## **Parte II - Exemplos em R**

---

## **Parte III - ML para classificação de imagens**

---

# Tipos de classificação

Existem dois grandes paradigmas de classificação de imagem (que também se aplicam ao machine learning):

- **Não-Supervisionada (*Unsupervised*)**: o algoritmo tenta ‘adivinar’ quem são as classes, e o usuário nomeia as classes *após* a classificação. Em ML e estatística é conhecido normalmente como *clustering*.
- **Supervisionada (*Supervised*)**: o usuário define as classes *antes* da classificação, seleciona amostras que representam cada classe, e *treina* o algoritmo a reconhecer as classes.

Esse último parece familiar...

O único passo adicional é transformar a imagem e as amostras em uma tabela:

Pixel	Banda 1	Banda 2	Banda 3	Classe
1	45	56	35	NA
2	67	45	23	NA
3	123	112	215	Solo
4	240	242	250	Urbano
5	12	16	8	Agua
6	67	32	56	NA

## Mais um algoritmo de ML: *Support Vector Machines*

Algoritmo popular para classificação de imagens de sensoriamento remoto

- Desenvolvido na década de 90 na AT&T Bell Labs por [Vladimir Vapnik](#)
- Ideias originais desenvolvidas na Rússia na décadas de 60-70

## Mais um algoritmo de ML: *Support Vector Machines*

Vantagens:

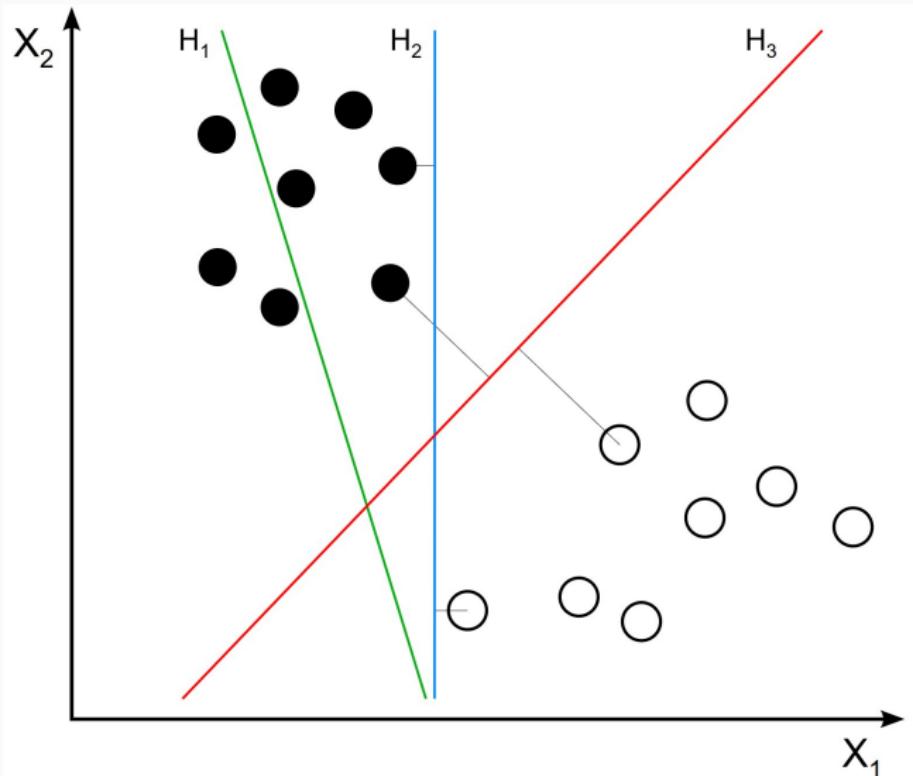
- Classificação e regressão
- Problemas Lineares e não lineares com o ‘truque do kernel’
- Eficiente em espaços multidimensionais in high dimensional spaces.
- Eficiente também quando o numero de variaveis é maior que o de amostras.
- Usa apenas um subconjunto das amostras de treino na solução, então eficiente em memória

## *Support Vector Machines (SVM)*

Princípio: o SVM tenta encontrar o melhor limiar de separação entre as classes.

- Para duas variáveis: uma linha
- Para três variáveis: um plano
- Para 4 ou mais variáveis: um *hiperplano*

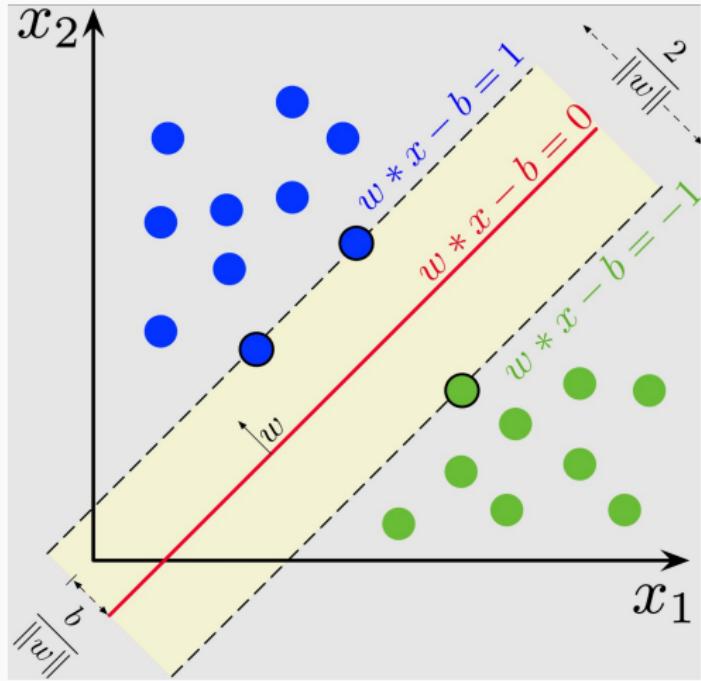
# *Support Vector Machines (SVM)*



By User:ZackWeinberg, based on PNG version by User:Cyc - This file was derived from: Svm separating hyperplanes.png, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=22877598>

# Support Vector Machines (SVM)

Para identificar o hiperplano, o SVM foca apenas nos casos limítrofes. Essas amostras são os *vetores de suporte*.

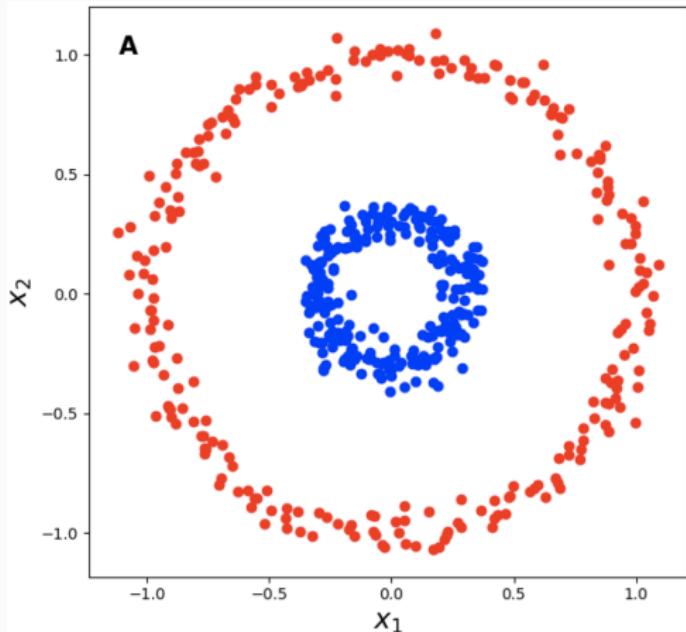


## Exemplo interativo

<https://greitemann.dev/svm-demo>

# *Support Vector Machines (SVM)*

Mas e se o problema não é linear?



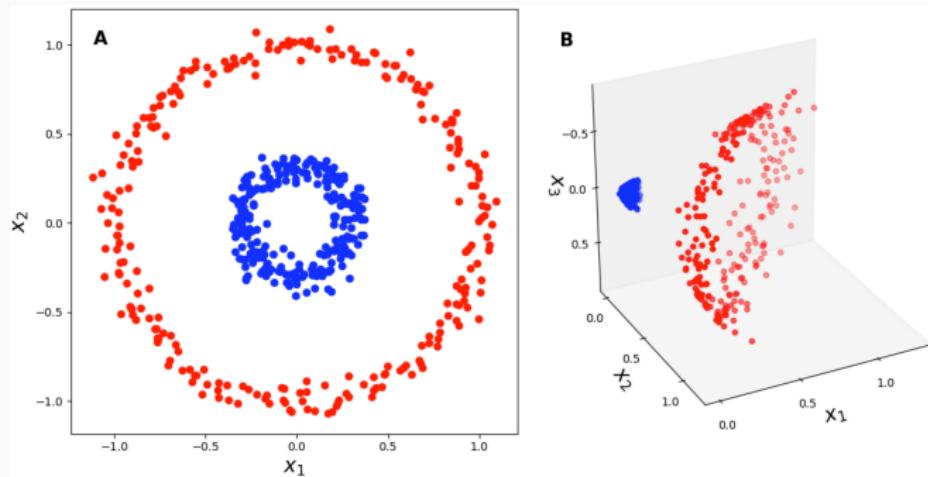
<https://gregorygundersen.com/blog/2019/12/10/kernel-trick/>

## *Support Vector Machines (SVM)*

Podemos aplicar transformações matemáticas que separem os dados em outras dimensões.

# *Support Vector Machines (SVM)*

Podemos aplicar transformações matemáticas que separem os dados em outras dimensões.



<https://gregorygundersen.com/blog/2019/12/10/kernel-trick/>

## *Support Vector Machines (SVM)*

O *truque do kernel* (*kernel trick*) é um método matemático que permite ao SVM ‘adivinar’ qual a melhor transformação para maximizar a separação dos dados. Diferentes funções de kernel podem ser utilizadas:

- **Linear:** apenas linhas/planos/hiperplanos. -**Polinomial:** permite curvas suaves.
- **Radial Basis Functions (RBF) / Gaussiano:** o mais comum, pois é extremamente flexível em se adaptar aos dados.

Existem outros, mas na prática o RBF é o mais usado.

<https://greitemann.dev/svm-demo>

# Parâmetros do SVM

São aprendidos pelo modelo.

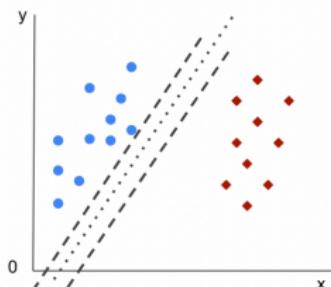
- Os vetores de suporte
- Os seus pesos (weights)
- O intercepto (viés)

# Hiperparâmetros do SVM

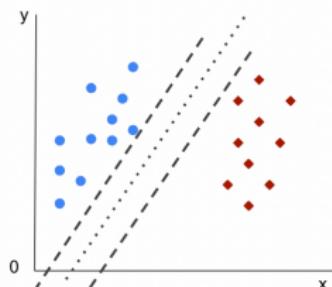
Precisam ser otimizados.

- Kernel: linear, quadrático, etc.
- Custo (cost): controla a amplitude da *margem*.

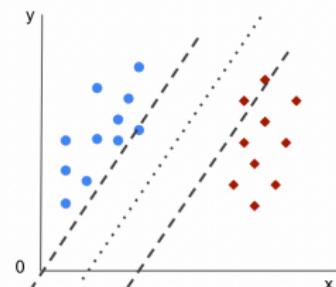
$C = 100$



$C = 10$



$C = 1$

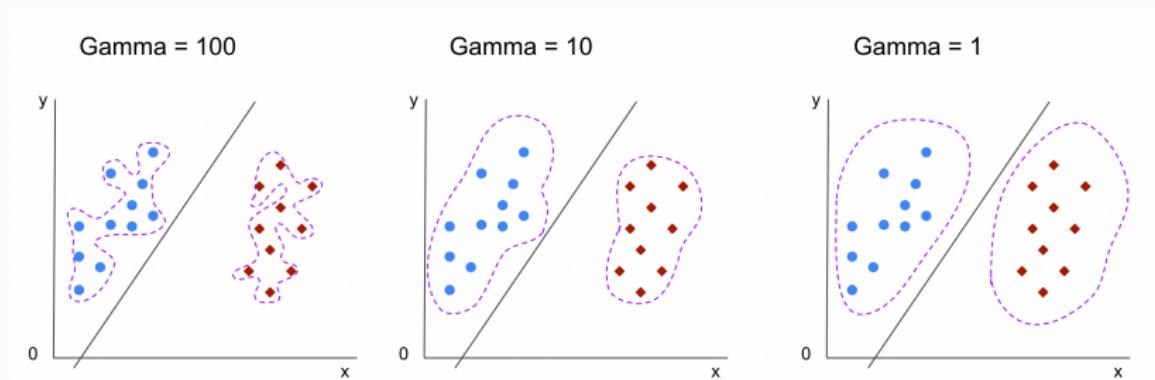


<https://stackabuse.com/understanding-svm-hyperparameters/>

# Hiperparâmetros do SVM

Precisam ser otimizados.

- Kernel: linear, quadrático, etc.
- Custo (cost): controla a amplitude da *margem*.
- Regularização ( $\gamma$  ou  $\sigma$ ): controla o ‘detalhamento’ das bordas de separação.



<https://stackabuse.com/understanding-svm-hyperparameters/>