Fundamentos de Machine Learning

Professora Stella Sposito
stella.sposito@facens.br

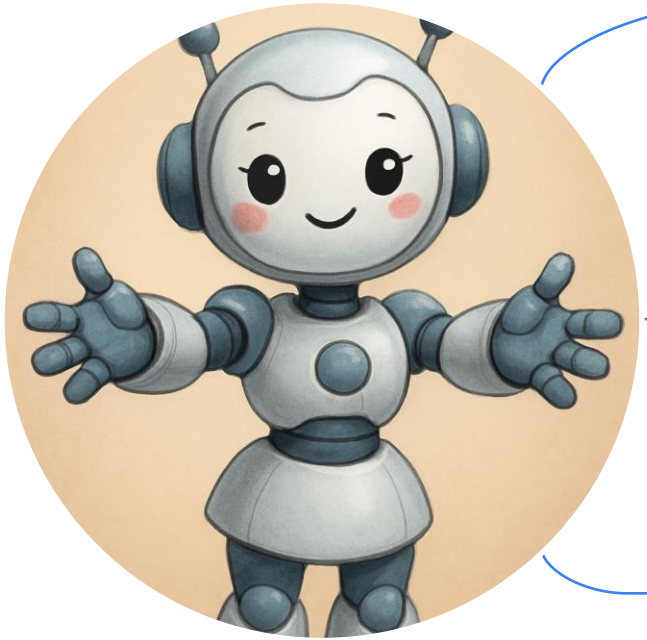
Quem sou eu?

Finalizado:
Graduada em Ciências Biológicas – UFSCar
Pós-Graduada em Ciências de Dados – Facens

Trabalhando:
Cientista de Dados - DP6 (Marketing Data)
Professora de Machine Learning - Facens

Em andamento:
Pós-Graduação em Cloud e Arquitetura de Dados - GRAN Faculdade
Mestrado em Sistemas de Informação - USP

Quem são vocês?



No que se formaram?

Estão trabalhando ou procurando por emprego ou dando uma pausa?

Estão fazendo outros cursos?

Ementa

- **Aula 1 – 08/11**

Aula Teórica + Atividade em aula + Atividade para casa

- **Aula 2 – 29/11**

Aula Teórica + Atividade em aula + Atividade para casa

- **Aula 3 – 06/12**

Aula Teórica + Atividade em aula + Atividade para casa

- **Aula 4 – 13/12**

Aula Teórica + Apresentação Final em aula

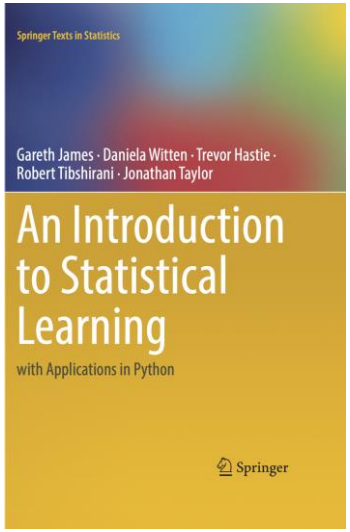
Notas:

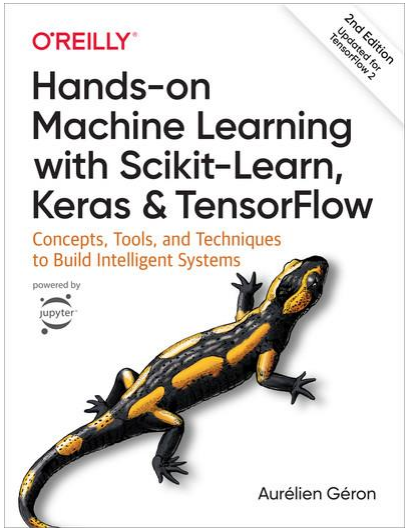
- Cada atividade vale **1 ponto** (Aula + Casa = **2 pontos**)
- Apresentação em aula vale **3 pontos**
- Participação em aula vale **1 ponto**

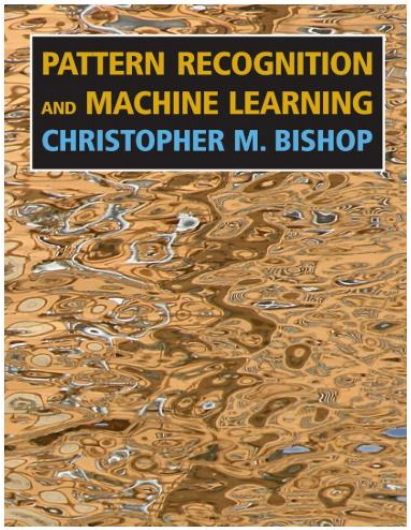
Presença:

- Aprovação com mínimo de 75% de presença
- Lista de chamada na parte da manhã e da tarde

Referências Bibliográficas







Outras Referências

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2. ed. New York: Springer, 2009.

PRADO, L. **Aprendizado de Máquina em Python**. São Paulo: Novatec, 2020.

MURPHY, K. P. **Probabilistic Machine Learning: An Introduction**. Cambridge: MIT Press, 2022.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997.

RASCHKA, S.; MIRJALILI, V. **Python Machine Learning**. 4. ed. Birmingham: Packt Publishing, 2022.

Machine Learning

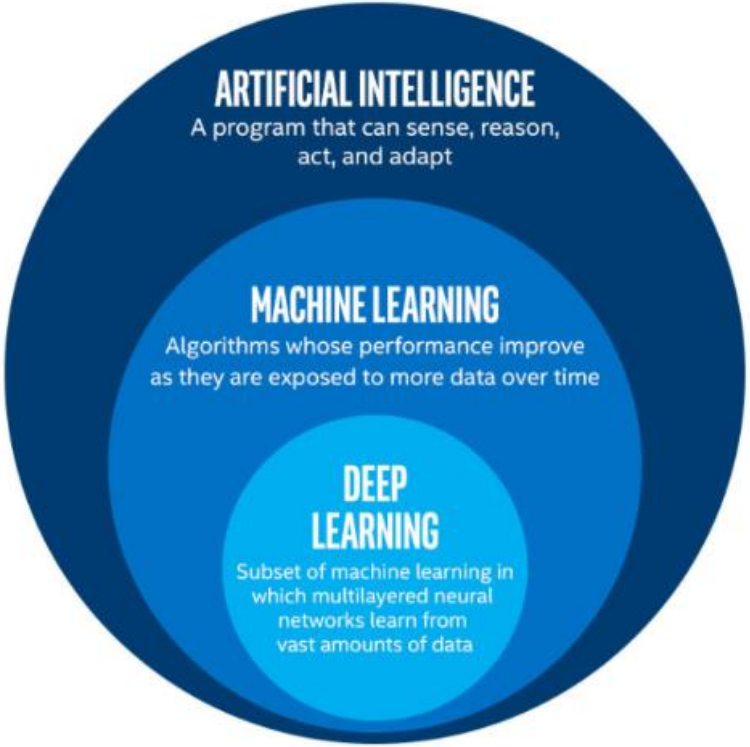
O que é Machine Learning?

- É um ramo da Inteligência Artificial.

- É a ciência que desenvolve algoritmos e modelos estatísticos para **ensinar as máquinas (com um treinamento)** a realizarem tarefas complexas.

- Os sistemas de computadores usam algoritmos de ML para **processar grandes quantidade de dados históricos, realizar previsões e identificar padrões.**

ML x IA


ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

O que é um modelo de ML?

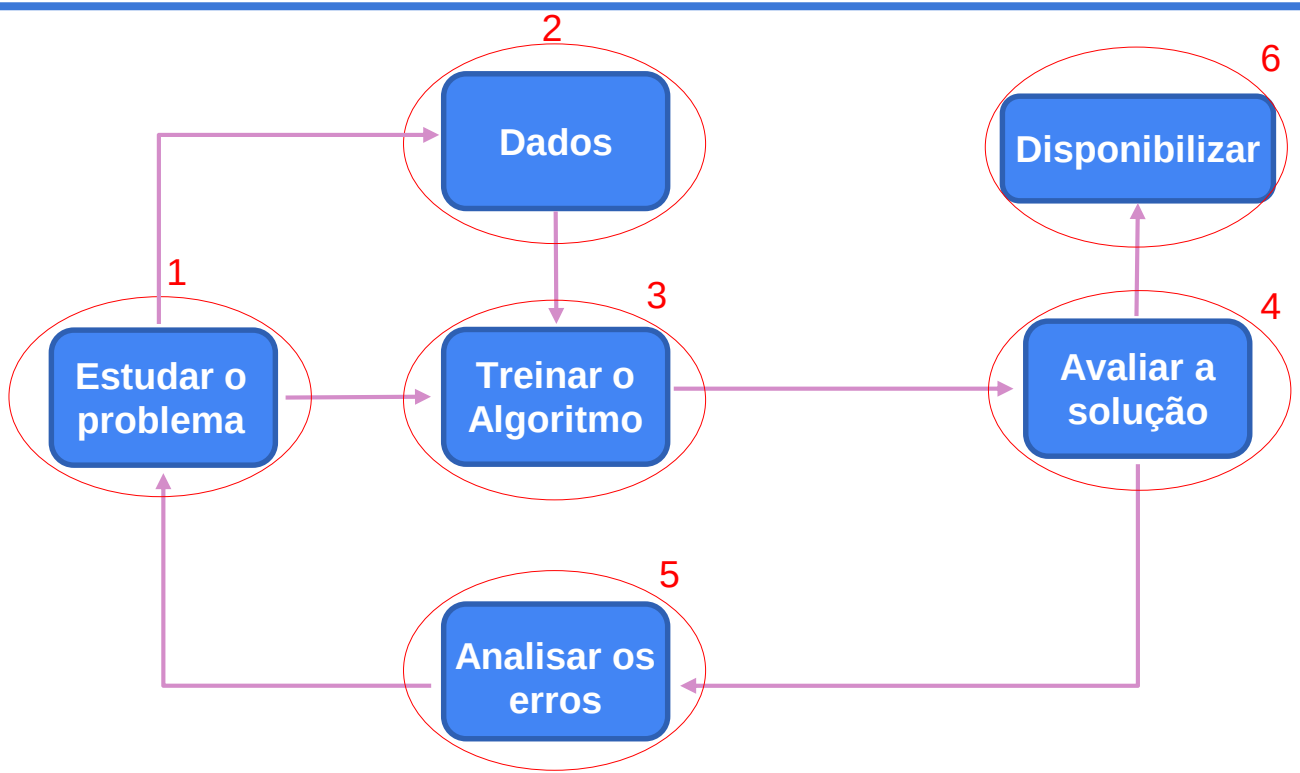
- É um modelo **matemático** projetado para identificar padrões e relações em um conjunto de dados para fazer previsões sobre novas informações.

- Os modelos de Machine Learning são desenvolvidos por meio de algoritmos e técnicas específicas para cada tipo de problema.

A finalidade de um modelo de ML é fazer previsões com a maior proximidade possível da realidade, ou seja, é **minimizar ao máximo a função de perda.**

Ferramentas matemáticas que quantificam a diferença entre as previsões do modelo e os valores reais dos dados.

Como funciona um modelo de ML?

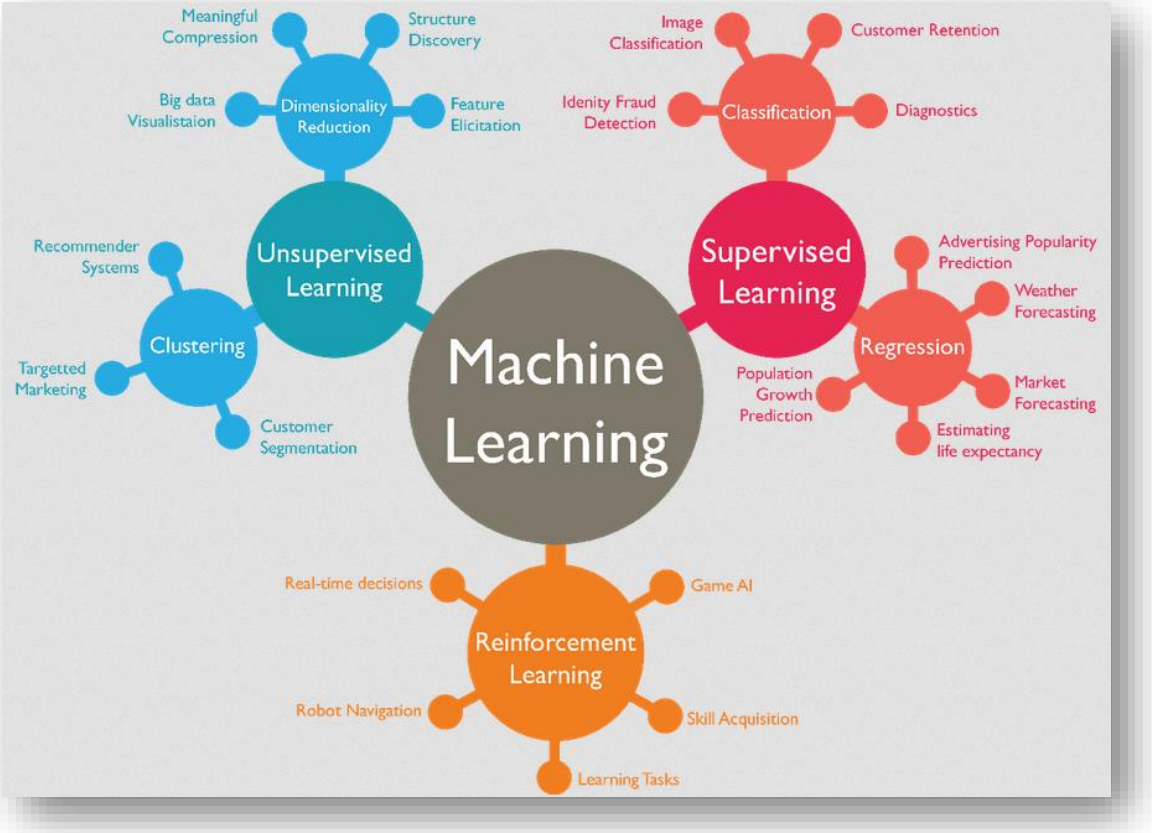


Exemplos de aplicações de modelos de ML

- Filtro de spam dos e-mails;
- Previsão do faturamento de uma empresa no próximo ano;
- Previsões de forecasting;
- Previsão de churn;
- Detecção de fraudes (tanto no cartão quanto fraudes por reconhecimento facial);
- Liberação de crédito para financiamentos;
- Segmentação de clientes com base em suas compras;
- Sistema de recomendação de filmes, produtos;
- Detecção de tumores a partir de exames de imagens;
- Classificação de textos de artigos, notícias;
- Sinalização de comentários ofensivos na internet, análise de sentimentos;
- Resumo automático de documentos;
- Criação de chatbots;
- Criação de bots inteligentes para jogos;

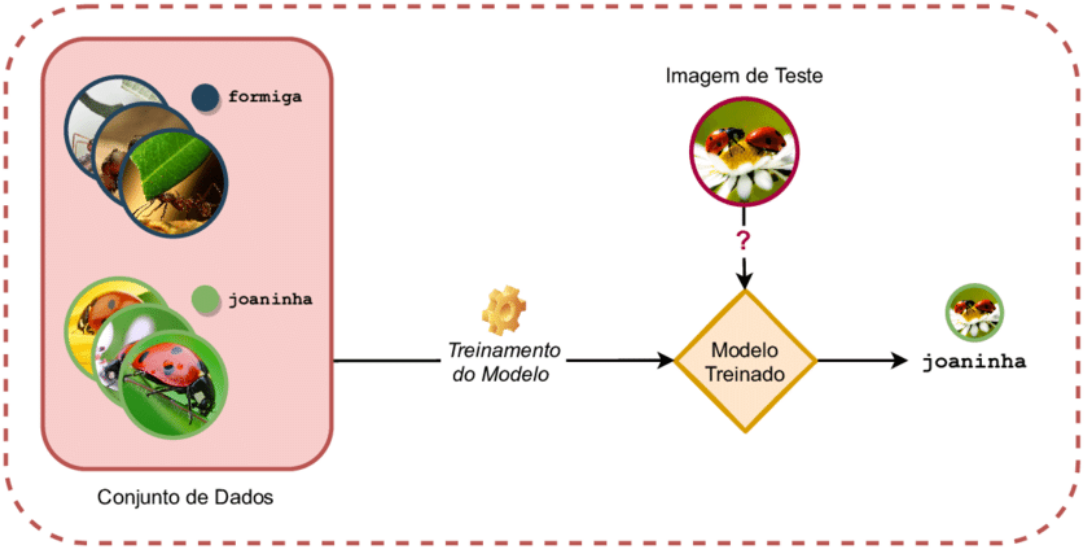
Tipos de Aprendizado

Tipos de Aprendizado

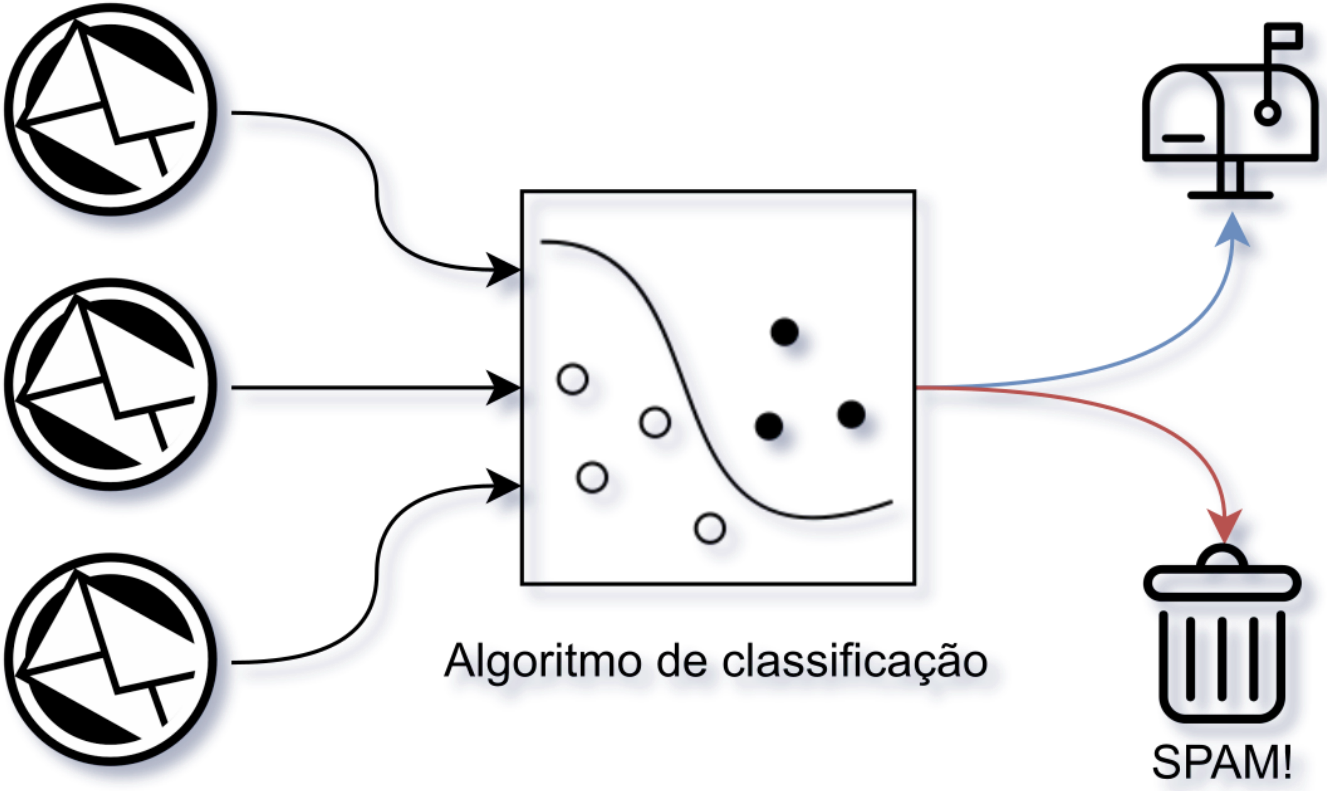


Aprendizado Supervisionado

- É um método de aprendizado de máquina em que os algoritmos aprendem com um treinamento baseado em rótulos (labels) fornecido previamente.
- “Supervisionado” pode ser visto como uma técnica de ensinar algoritmos com uma supervisão humana.



Aprendizado Supervisionado



Tipos de Aprendizado Supervisionado

Classificação

- Quando quero que o modelo classifique novos dados com base em dados já rotulados.
- Por exemplo, classificar novos e-mails como spam ou não spam.

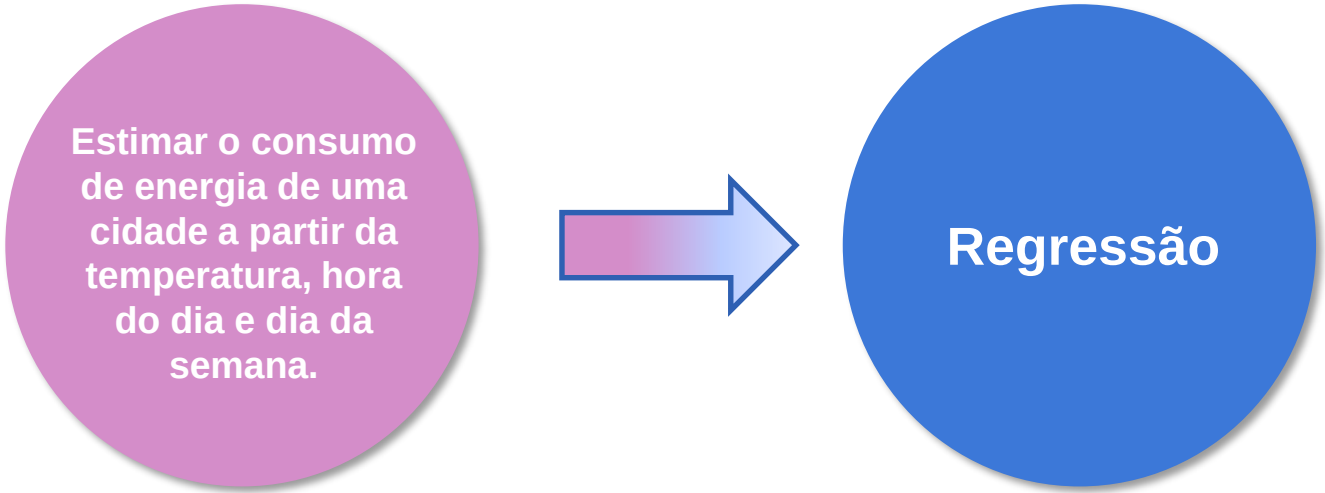
Árvores de Decisão, KNN, Regressão Logística, Random Forest, XGBoost, etc.

Regressão

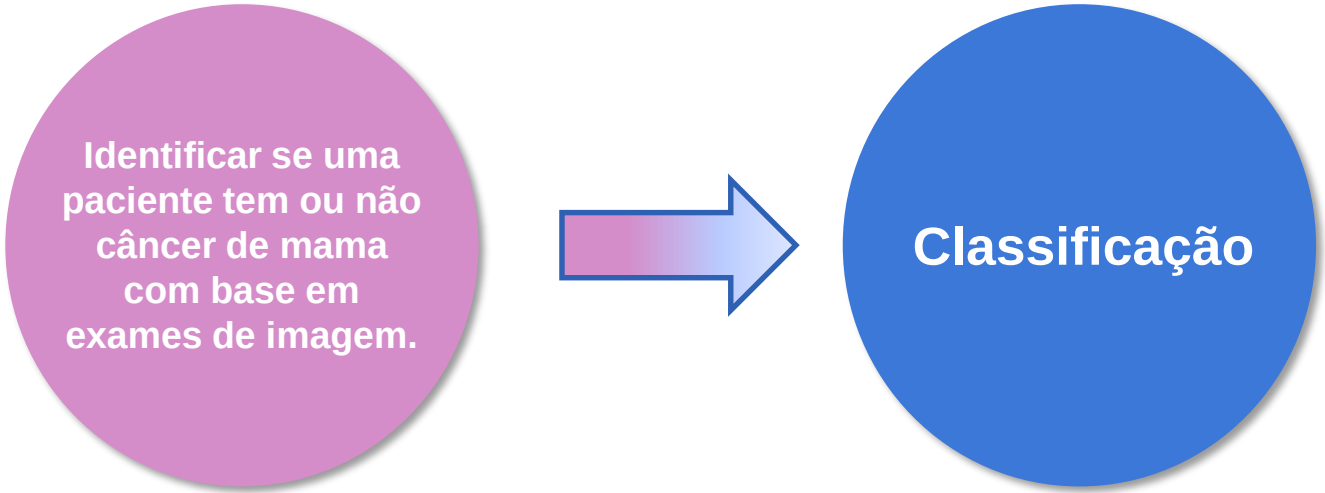
- Quando quero que o preveja um valor numérico dado um número de características.
- Por exemplo, quero prever o valor de uma casa com base nas informações de área, cômodos, localização.

Regressão Linear, Random Forest Regressor outros Regressors.

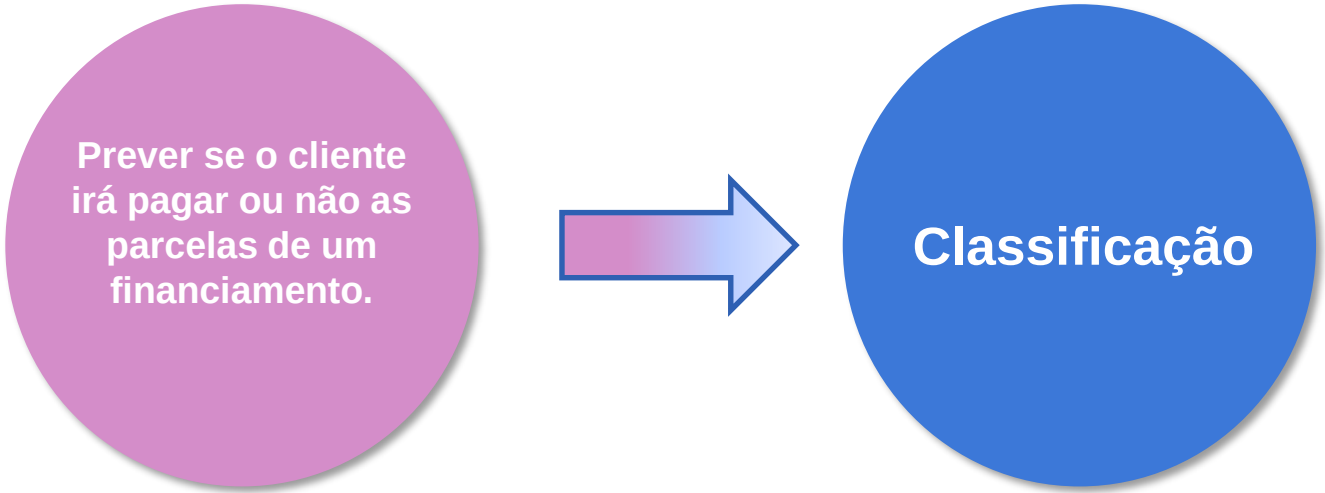
Classificação Vs Regressão



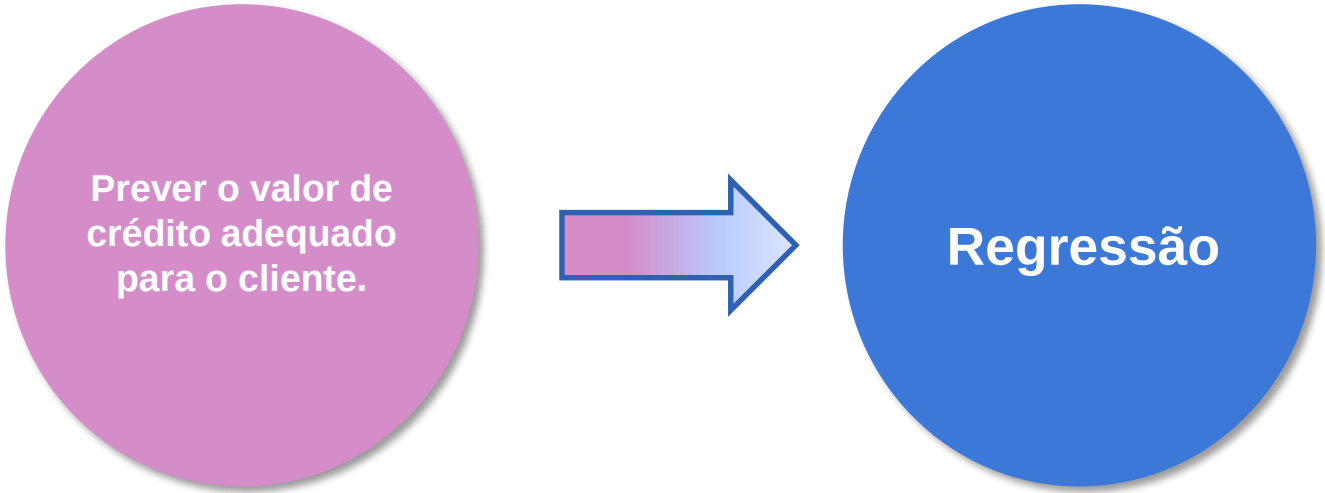
Classificação Vs Regressão



Classificação Vs Regressão

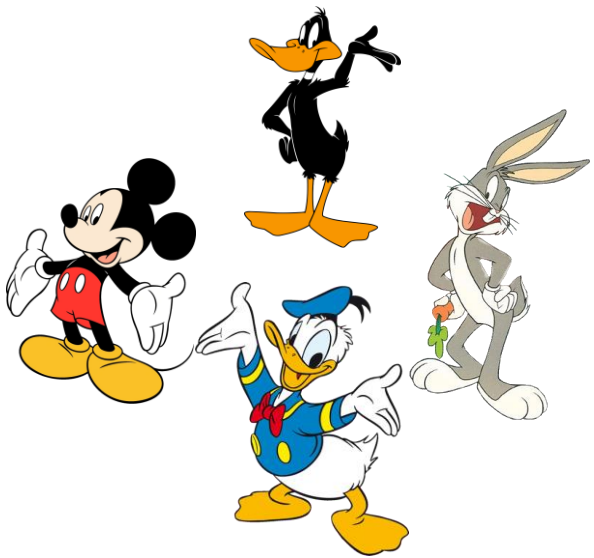


Classificação Vs Regressão



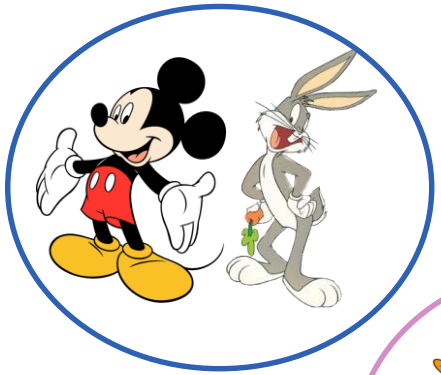
Aprendizado Não Supervisionado

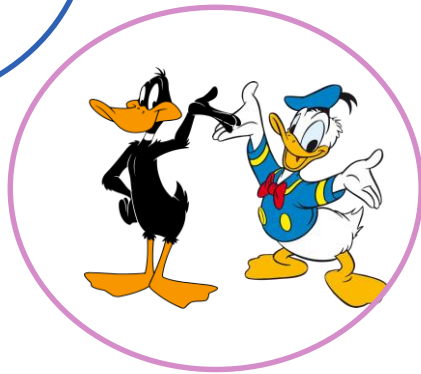
- Como o nome já diz, “Não Supervisionado” não tem supervisão humana, portanto, não existem rótulos para os dados de treinamento.
- Neste caso, o sistema tenta aprender sozinho e encontrar padrão nos dados com base nas informações fornecidas.



Aprendizado Não Supervisionado

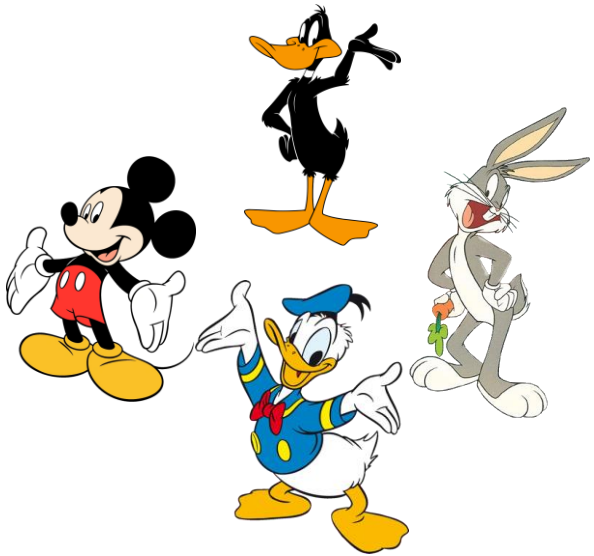
Algoritmo de Clusterização





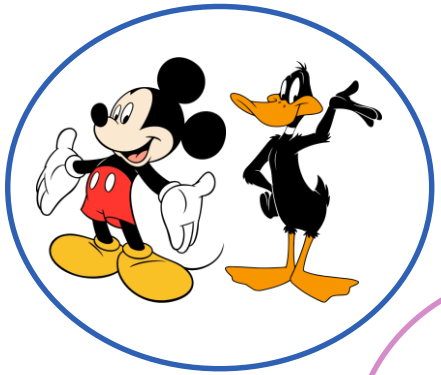
Aprendizado Não Supervisionado

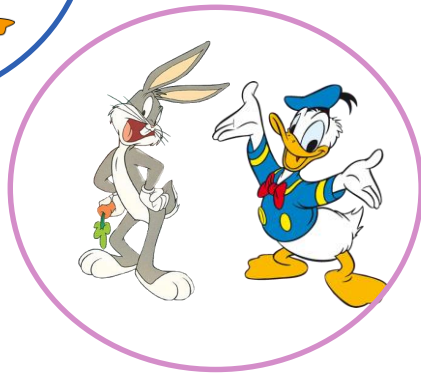
- Como o nome já diz, “Não Supervisionado” não tem supervisão humana, portanto, não existem rótulos para os dados de treinamento.
- Neste caso, o sistema tenta aprender sozinho e encontrar padrão nos dados com base nas informações fornecidas.



Aprendizado Não Supervisionado

Algoritmo de Clusterização

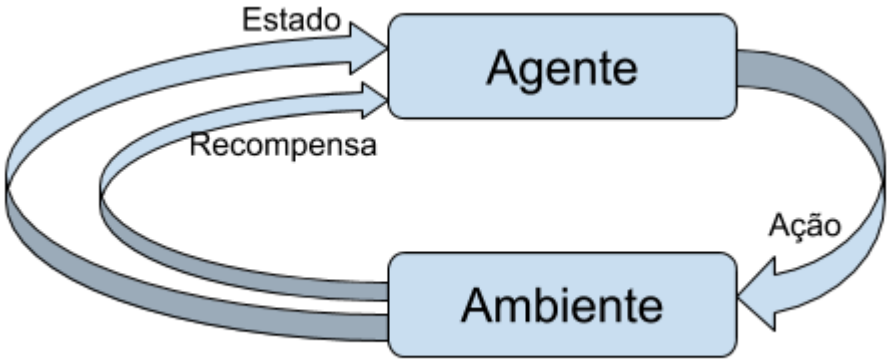




Aprendizado por Reforço

- **Agente:** cachorro
- **Ambiente:** quintal
- **Estado:** posição do cachorro e da bolinha
- **Ação:** andar, correr, pegar a bolinha
- **Recompensa:** petisco quando pega a bolinha

O cachorro aprende **experimentando ações e recebendo recompensas**, ajustando o comportamento para **pegar a bolinha mais rápido no futuro**.



Como construir um modelo de ML?

Análise Exploratória/Processamento dos Dados

Análise Exploratória (EDA)

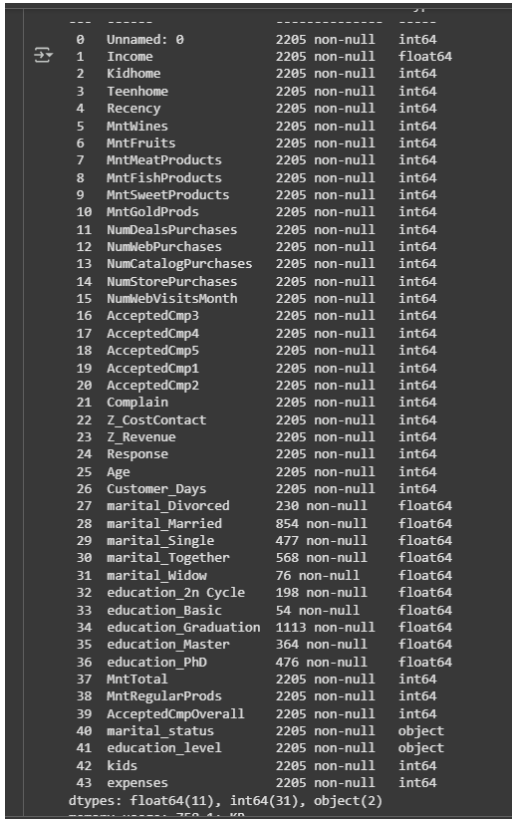
- Antes de pensarmos em construir um modelo, é necessário sabermos com quais dados estamos lidando.

- A análise exploratória equivalente à primeira etapa de **estudar o problema e entender os dados.**

- A análise exploratória envolve as seguintes etapas:

Entendimento das colunas – entendimento dos tipos de dados – resumos estatísticos – visualizações (histogramas, boxplots, correlações) – identificação de outliers – identificação de colunas relevantes/não relevantes

Entendimento das colunas e tipos de dados



```
---  ------                  --------------  -----
 0   Unnamed: 0              2205 non-null   int64
 1   Income                  2205 non-null   float64
 2   Kidhome                 2205 non-null   int64
 3   Teenhome                2205 non-null   int64
 4   Recency                 2205 non-null   int64
 5   MntWines                2205 non-null   int64
 6   MntFruits               2205 non-null   int64
 7   MntMeatProducts         2205 non-null   int64
 8   MntFishProducts         2205 non-null   int64
 9   MntSweetProducts        2205 non-null   int64
 10  MntGoldProds            2205 non-null   int64
 11  NumDealsPurchases       2205 non-null   int64
 12  NumWebPurchases         2205 non-null   int64
 13  NumCatalogPurchases     2205 non-null   int64
 14  NumStorePurchases       2205 non-null   int64
 15  NumWebVisitsMonth       2205 non-null   int64
 16  AcceptedCmp3            2205 non-null   int64
 17  AcceptedCmp4            2205 non-null   int64
 18  AcceptedCmp5            2205 non-null   int64
 19  AcceptedCmp1            2205 non-null   int64
 20  AcceptedCmp2            2205 non-null   int64
 21  Complain                2205 non-null   int64
 22  Z_CostContact           2205 non-null   int64
 23  Z_Revenue               2205 non-null   int64
 24  Response                2205 non-null   int64
 25  Age                     2205 non-null   int64
 26  Customer_Days           2205 non-null   int64
 27  marital_Divorced        230 non-null    float64
 28  marital_Married         854 non-null    float64
 29  marital_Single          477 non-null    float64
 30  marital_Together        568 non-null    float64
 31  marital_Widow           76 non-null     float64
 32  education_2n Cycle      198 non-null    float64
 33  education_Basic         54 non-null     float64
 34  education_Graduation    1113 non-null   float64
 35  education_Master        364 non-null    float64
 36  education_PhD           476 non-null    float64
 37  MntTotal                2205 non-null   int64
 38  MntRegularProds         2205 non-null   int64
 39  AcceptedCmpOverall      2205 non-null   int64
 40  marital_status          2205 non-null   object
 41  education_level         2205 non-null   object
 42  kids                    2205 non-null   int64
 43  expenses                2205 non-null   int64
dtypes: float64(11), int64(31), object(2)
```

- Quais dessas colunas serão úteis?
- Existem colunas que eu posso remover?
- O tipo de cada coluna está correto para a minha análise?

Resumos Estatísticos

```
#com o describe, conseguimos analisar a tendência central e variação através da contagem da média e desvio padrão
df.describe()
```

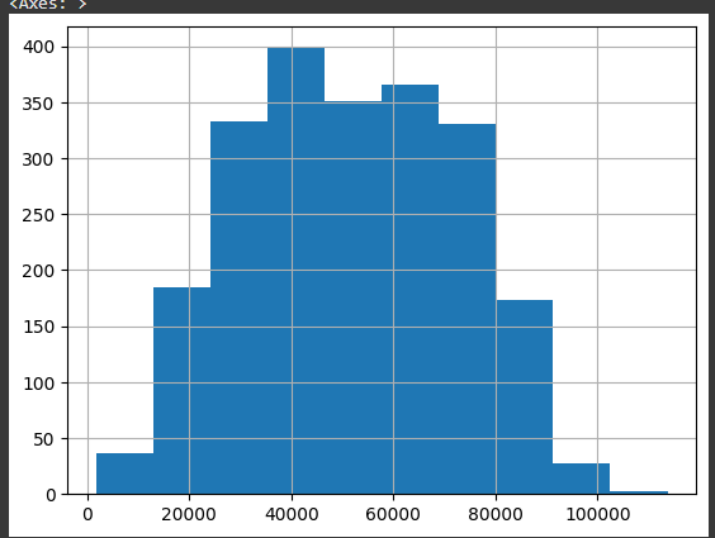
	Unnamed: 0	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	...
count	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000	...
mean	1102.000000	51622.094785	0.442177	0.506576	49.009070	306.164626	26.403175	165.312018	37.756463	27.128345	...
std	636.672993	20713.063826	0.537132	0.544380	28.932111	337.493839	39.784484	217.784507	54.824635	41.130468	...
min	0.000000	1730.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
25%	551.000000	35196.000000	0.000000	0.000000	24.000000	24.000000	2.000000	16.000000	3.000000	1.000000	...
50%	1102.000000	51287.000000	0.000000	0.000000	49.000000	178.000000	8.000000	68.000000	12.000000	8.000000	...
75%	1653.000000	68281.000000	1.000000	1.000000	74.000000	507.000000	33.000000	232.000000	50.000000	34.000000	...
max	2204.000000	113734.000000	2.000000	2.000000	99.000000	1493.000000	199.000000	1725.000000	259.000000	262.000000	...

8 rows × 42 columns

Visualizações (Análise Univariada e Bivariada)

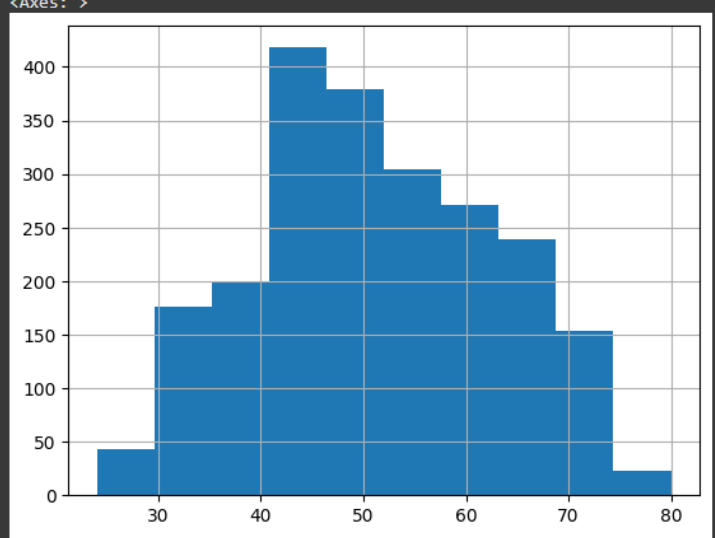
```
df['Income'].hist()
```

```
<Axes: >
```




```
df['Age'].hist()
```

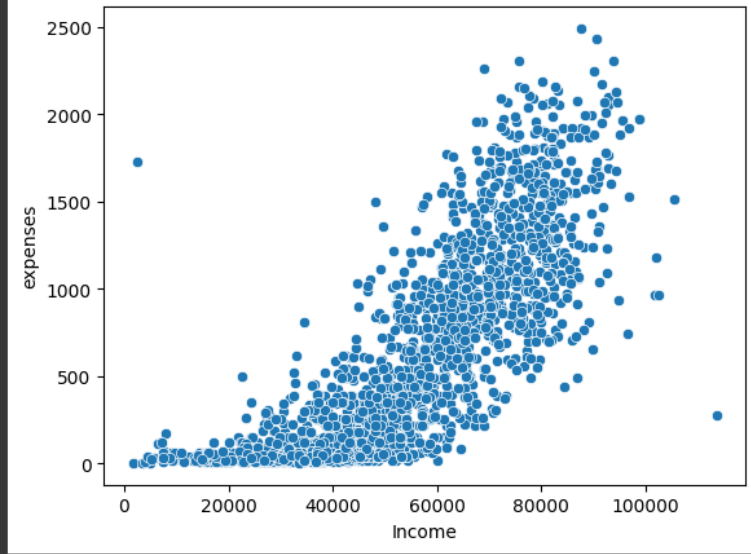
```
<Axes: >
```



Visualizações (Análise Univariada e Bivariada)

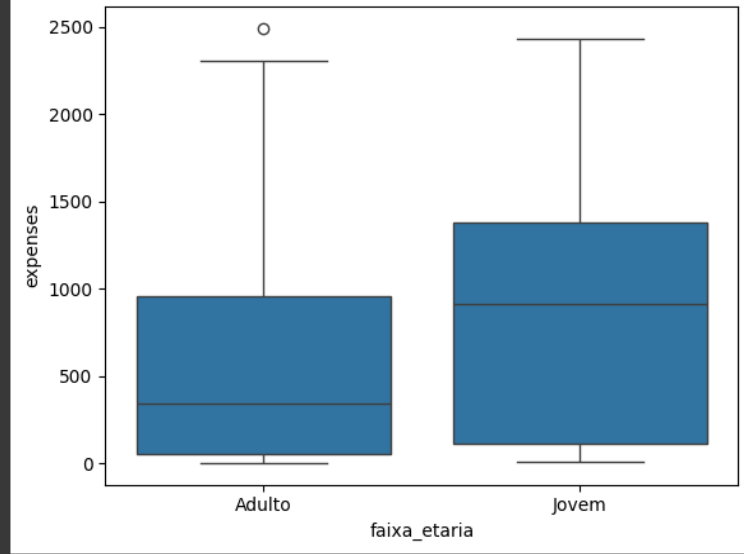
```
sns.scatterplot(x='Income', y='expenses', data=df)
```

```
<Axes: xlabel='Income', ylabel='expenses'>
```




```
sns.boxplot(x='faixa_etaria', y='expenses', data=df)
```

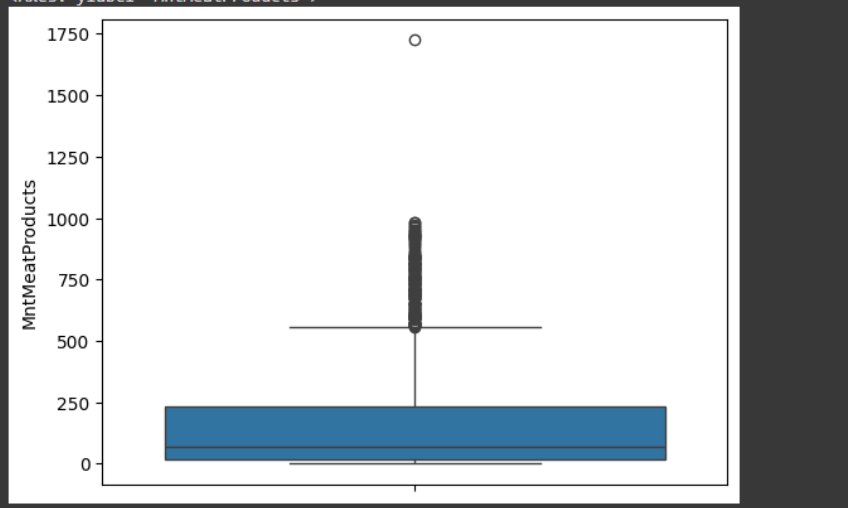
```
<Axes: xlabel='faixa_etaria', ylabel='expenses'>
```

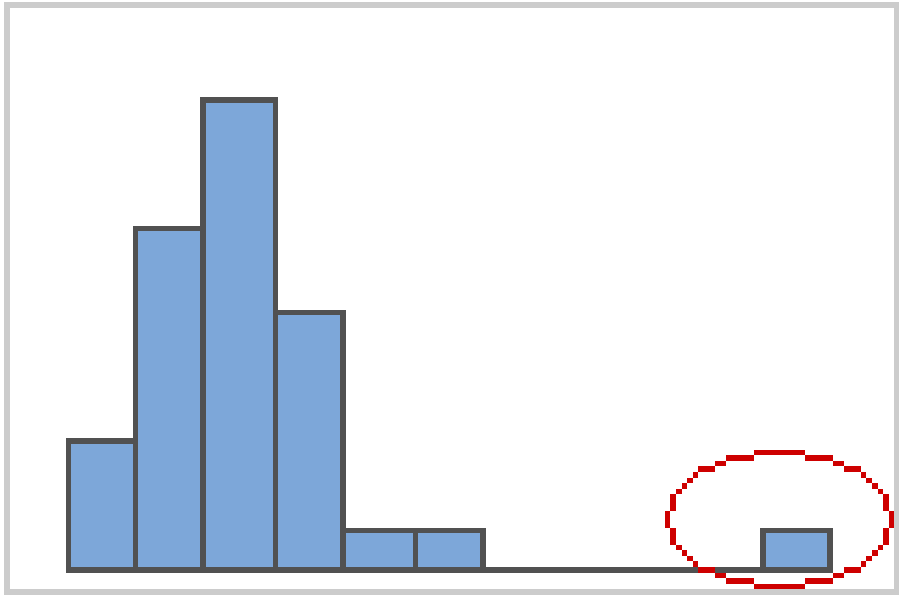


Outliers

```
sns.boxplot(data=df['MntMeatProducts'])
```

```
<Axes: ylabel='MntMeatProducts'>
```





Processamento dos Dados

- Com o entendimento dos nossos dados, agora precisamos tratá-los. Podemos dividir o processamento dos dados nas seguintes etapas:

Limpeza de dados – tratamento de dados nulos – tratamento de valores categóricos – tratamento de valores numéricos – remoção de duplicatas – remoção de outliers – criação de novas variáveis

Limpeza - Tratamento de dados nulos

```
#verificação de valores nulos
df.isnull().sum()
```

```
Airline Name                    0
Overall_Rating                  0
Review_Title                    0
Review Date                     0
Review                          0
Aircraft                    12037
Type Of Traveller            1749
Seat Type                     241
Route                        1826
Date Flown                   1758
Seat Comfort                 2246
Cabin Staff Service          2336
Food & Beverages             5778
Ground Service               2646
Inflight Entertainment       8471
Wifi & Connectivity         12459
dtype: int64
```

Missing not at random – ocorre quando o motivo de um valor estar ausente é devido ao próprio valor em si.

Missing at random – ocorre quando o valor ausente é devido outra variável.

Missing completely at random – ocorre quando não há um padrão na ausência do valor.

Limpeza - Tratamento de dados nulos

In [7]:
```
#porcentagem de nulos
porcentagem = (df.isnull().sum()/df.shape[0]).sort_values(ascending= True) * 100
df_nulos = pd.DataFrame(porcentagem, columns=['% de Nulos'])
df_nulos = df_nulos.sort_values(by='% de Nulos', ascending=False)
df_nulos
```

Out[7]:

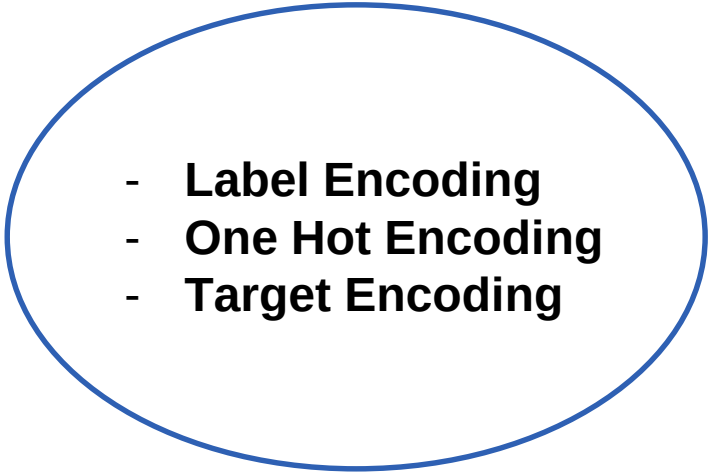
	% de Nulos
Wifi & Connectivity	70.381878
Aircraft	67.997966
Inflight Entertainment	47.853350
Food & Beverages	32.640380
Ground Service	14.947464
Cabin Staff Service	13.196249
Seat Comfort	12.687832
Route	10.315219
Date Flown	9.931081
Type Of Traveller	9.880240
Seat Type	1.361428
Airline Name	0.000000
Overall_Rating	0.000000
Review_Title	0.000000
Review Date	0.000000
Review	0.000000

O que fazer com esses dados nulos?

- Remover
- Substituir (média, mediana ou moda)
- Criar uma nova variável booleana

Tratamento de valores categóricos - encoders

- Alguns algoritmos mais simples não conseguem lidar com variáveis categóricas, portanto, é necessário transformá-las em numéricas.



Label Encoder

- Transforma variáveis categóricas em variáveis numéricas, porém, cada categoria se transforma em um número único, criando uma ordem artificial (hierarquia) entre as categorias.

Cor	Cor_encoded
Vermelho	2
Azul	0
Verde	1
Azul	0
Vermelho	2

Ou seja:

- Azul → 0
- Verde → 1
- Vermelho → 2

Vantagem – direta, não cria novas colunas

Desvantagem – pode ser mal interpretada por ordem de importância

One Hot Encoder

- Esta estratégia converte cada valor de categoria em uma nova coluna, atribuindo valor 0 ou 1 à coluna.

Cor	Azul	Verde	Vermelho
Azul	1	0	0
Verde	0	1	0
Vermelho	0	0	1
Azul	1	0	0

Vantagem – eficaz, não é mal interpretada
Desvantagem – dimensionalidade

Target Encoder

- Esta estratégia olha para o target (variável que está querendo prever) e calcula a média dele para cada categoria.

- Por exemplo, se você está prevendo o preço de casas e a cor da casa é a variável categórica, o Target Encoder vai calcular a média do preço de todas as casas vermelhas, azuis, etc, e substitui cada valor categórico pela média do alvo correspondente.

Vantagem – ajuda a entender a relação entre a variável de previsão e a variável categórica, evita dimensionalidade
Desvantagem – overfitting

Tratamento de valores numéricos - scalers

- Alguns algoritmos baseados em distância entre registros podem ser impactados por intervalos entre valores ou escalas diferentes. Por exemplo:

Idade: varia de 18 a 80 anos
Renda anual: varia de R$ 20.000 a R$ 500.000

Se você deixar os dados crus, a **renda** vai dominar o modelo, porque tem valores muito maiores que a idade.

Solução: **padronização**, coloca as duas variáveis na mesma escala (média 0, desvio 1), evitando que a renda domine o modelo.

Tratamento de valores numéricos - scalers

- Uma outra alternativa para transformação de dados numéricos é a **Normalização**.

- Na normalização, os valores são redimensionados para um intervalo fixo (normalmente entre 0 e 1, mas pode ser -1 e 1, por exemplo)

- Enquanto a padronização (StandardScaler) é mais recomendada para distribuições normais, a normalização é mais recomendada para algoritmos baseados em distância (KNN, K-means, etc)

Remoção de duplicatas

- Dados duplicados podem aparecer e precisam ser removidos para não gerarem inconsistências nas previsões.

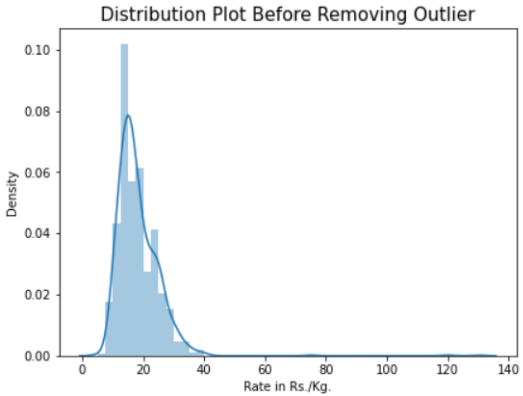
- Eles ocorrem quando duas ou mais linhas apresentam exatamente os mesmos valores para todas as colunas.

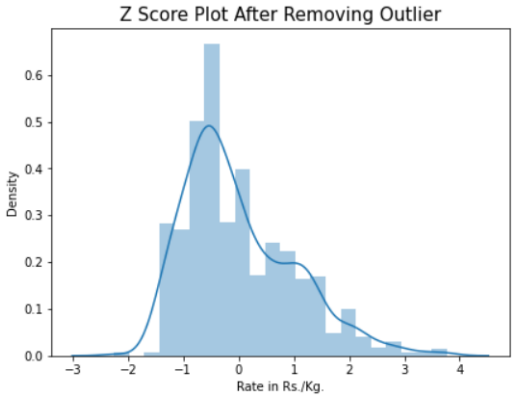
Em pandas, podemos verificas os casos duplicados assim:

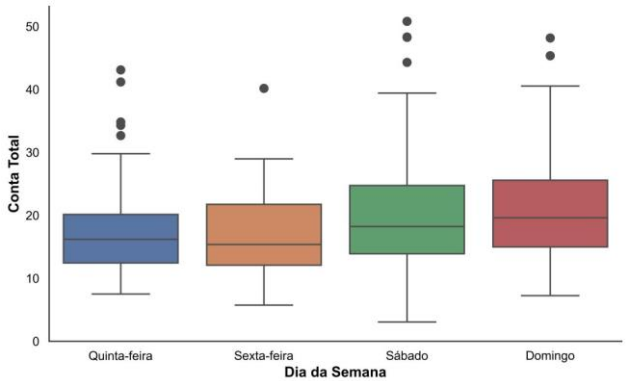
```
df.duplicated()
df.duplicated(subset=['col1', 'col2'])
```

Remoção de outliers

- Para verificar a existência de outliers, podemos analisar a distribuição dos dados por meio de um **histograma**, ou analisando os valores em cada quartil de um **boxplot**.
- Antes de sair removendo outliers, precisamos entender se é um dado realmente errado (ou se tem algum impacto na análise como um todo), ou se é um dado real que não pode ser removido, ou seja, ele nos traz alguma informação valiosa.







Criação de Novas Variáveis (Feature Engineering)

- É a parte de aumentar a inteligência dos dados. Consiste em uma etapa mais criativa e estratégica para criar ou transformar variáveis para aumentar a capacidade preditiva do modelo.

- Exemplos:

Criar novas colunas derivadas (idade = data_atual – **data_nascimento**)

Combinar variáveis (renda_per_capita = **renda/ nºpessoas**)

Criar variáveis temporais (dia da semana, mês, hora do dia, feriado, etc)

Agrupar categorias pouco frequentes em um valor “outros”

Antes de criarmos o modelo então...

Divisão em treino, teste e validação

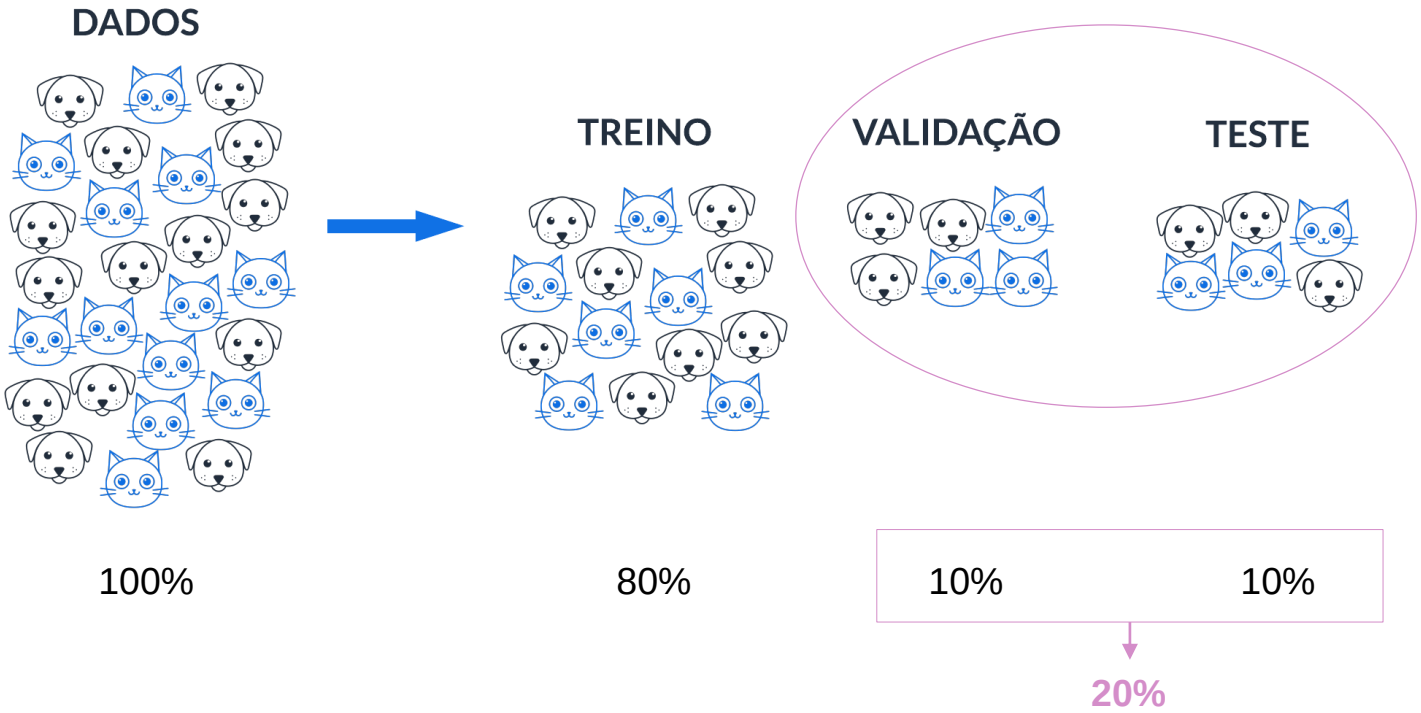
Dando início na modelagem

Imagine que temos um dataset de 20.000 linhas e 20 colunas, sendo uma delas a que queremos prever (target)

Para rodar algum modelo, precisamos primeiro treiná-lo.

Devemos treinar o modelo com todos os nossos dados?
As 20k linhas e 20 colunas?

Dando início na modelagem



Conjunto de Validação x Conjunto de Teste

- Se treinarmos com todos os dados, o modelo vai aprender tudo, mas não será testado, então sua performance com dados desconhecidos será desconhecida para nós também.
- Por isso há a necessidade dessa divisão para os dados de teste.
- Além do conjunto de teste, há o conjunto de validação, que, em alguns casos, é extremamente útil para melhorar o desempenho do modelo.

Conjunto de Validação

Usado para ajustar hiperparâmetros e para comparar modelos diferentes. Normalmente, o conjunto de validação pode ser criado para o método **Holdout** e na **Validação Cruzada**.

Conjunto de Teste

Usado somente no final, após escolher o modelo e os hiperparâmetros. Os dados de teste simulam dados nunca vistos antes pelo modelo e servem para medir a generalização real.

Método Holdout

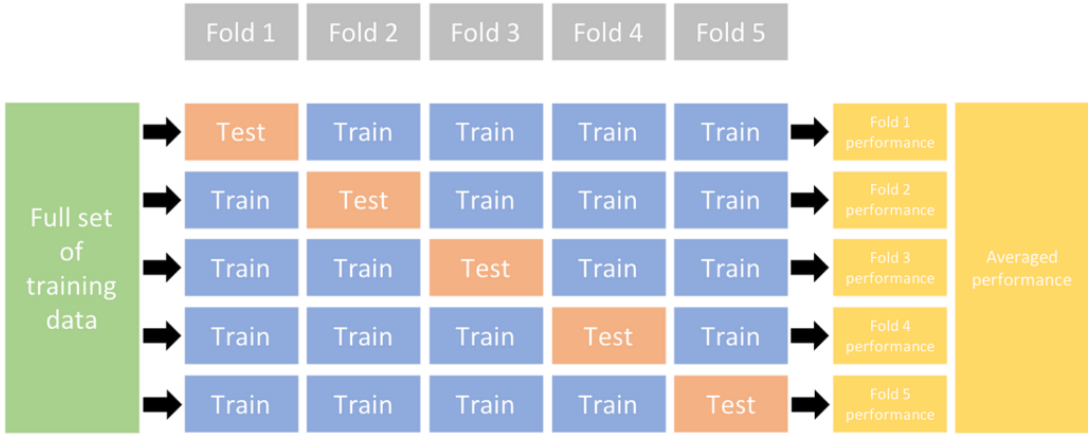
- É o método utilizado para dividir o conjunto de dados em **dados de treinamento e teste**.
- O conjunto de validação pode ser incluído neste método de forma manual, mas o método holdout divide os dados apenas em duas partes.
- Podemos dividir os dados com **pandas** ou com a biblioteca **Scikit-Learn**.

```python
import pandas as pd

# Suponha que df é o seu dataframe completo
# Ex.: df = pd.read_csv("dados.csv")

# 1. Embaralhar o dataframe
df = df.sample(frac=1, random_state=42).reset_index(drop=True)

# 2. Definir o tamanho do treino
train_size = int(0.8 * len(df))  # 80% para treino

# 3. Criar os conjuntos de treino e teste
df_train = df.iloc[:train_size]
df_test = df.iloc[train_size:]

print("Treino:", df_train.shape)
print("Teste:", df_test.shape)
```

```python
from sklearn.model_selection import train_test_split

# Suponha que X = features e y = target

# Divide 80% treino e 20% teste
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print("Treino:", X_train.shape, y_train.shape)
print("Teste:", X_test.shape, y_test.shape)
```

Validação Cruzada

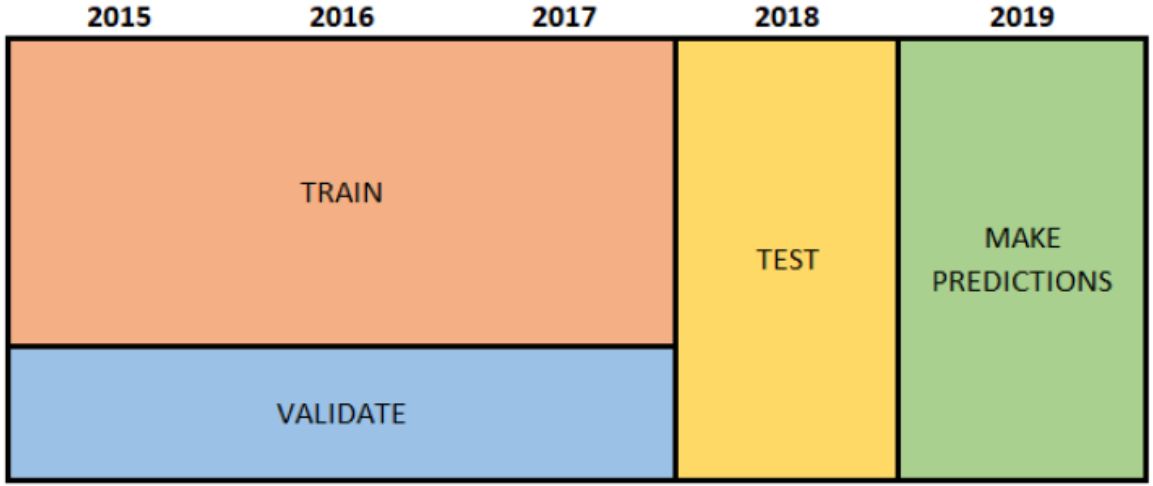
- É uma técnica muito utilizada para avaliação de desempenho de modelos de aprendizado de máquina com **o conjunto de treinamento**.
- A utilização de Cross Validation (CV) tem altas chances de detectar se o modelo poderá ou não ter overfitting.
- Existem alguns métodos para realizar a Validação Cruzada, mas vamos focar no **K-fold.**



Validação Out-of-time

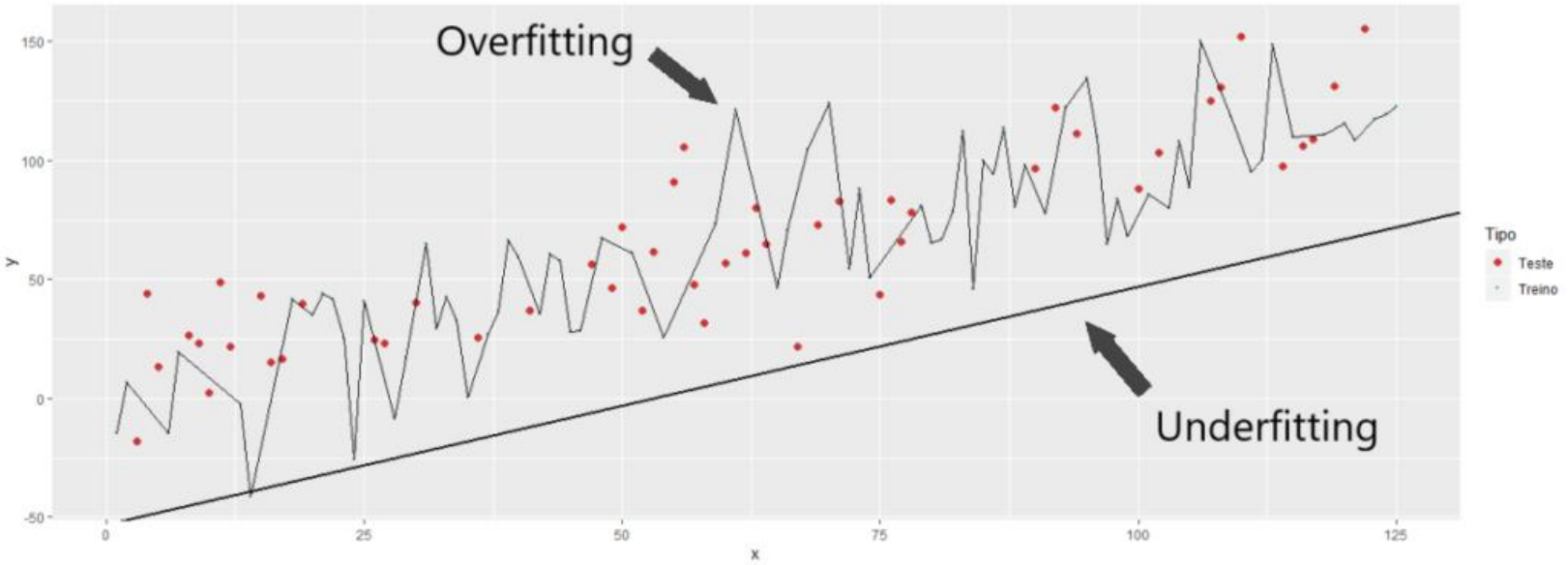
- É uma técnica para treinar seu modelo com os primeiros X meses e validar com os últimos Y meses, evitando **vazamento de dados.**

- Isso é importante para verificar se o modelo generaliza bem mesmo em amostrar que estão em tempos diferentes do que foram treinados. Muito útil para problemas reais.



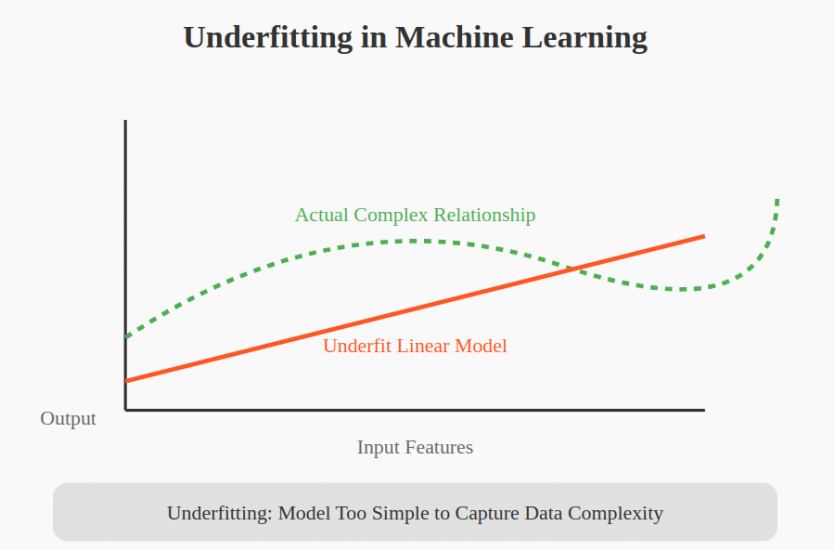
Overfitting e Underfitting

Overfitting x Underfitting



Underfitting

- Acontece quando o desempenho do modelo já é ruim no próprio treinamento.
- O modelo não consegue encontrar relações entre as variáveis, portanto, não aprendeu bem no treinamento e consequentemente não terá boa performance no teste.



Underfitting

Características

O modelo tem desempenho ruim tanto em conjuntos de dados de treinamento quanto de teste.

Não consegue capturar a complexidade do conjunto de dados.

Adicionar mais dados de treinamento não melhora significativamente o desempenho.

Isso ocorre quando os modelos são muito simples (por exemplo, usando regressão linear para um conjunto de dados não linear).

Como consertar?

Aumente a complexidade do modelo usando modelos mais complexos.

Adicione mais features relevantes para aprimorar o aprendizado.

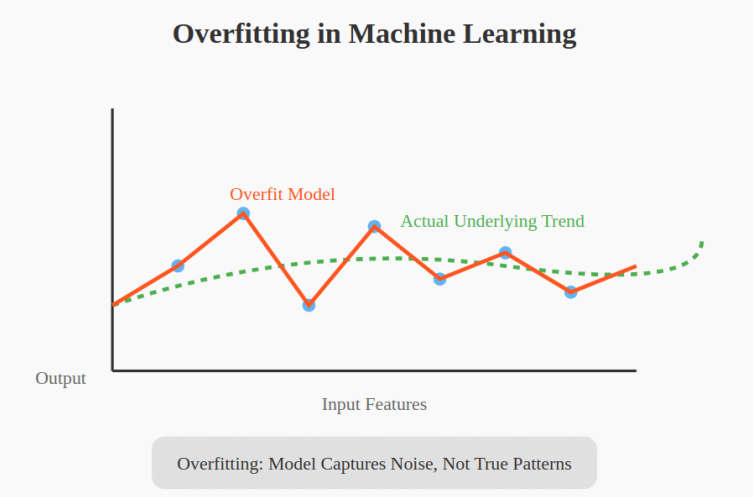
Reduza a regularização.

Treine por mais tempo (em mais épocas)

Overfitting

- Acontece quando o modelo tem um excelente desempenho nos dados de treino, mas uma péssima performance com os dados de teste.

- Podemos entender que o modelo aprendeu tão bem as relações existentes no treino, que acabou decorando o que deveria ser feito, então, ao receber dados nunca vistos antes, não soube encontrar padrões. Neste cenário, o modelo não tem capacidade de generalização.



Overfitting

Características

O modelo tem um desempenho excepcionalmente bom em dados de treinamento, mas ruim em dados de teste.

Ele captura ruídos e anomalias em vez de padrões gerais.

O modelo tem muitos parâmetros e é excessivamente complexo.

Não é possível generalizar bem para novos conjuntos de dados.

Como consertar?

Técnicas para aumentar regularização.

Validação cruzada antes de testar os modelos.

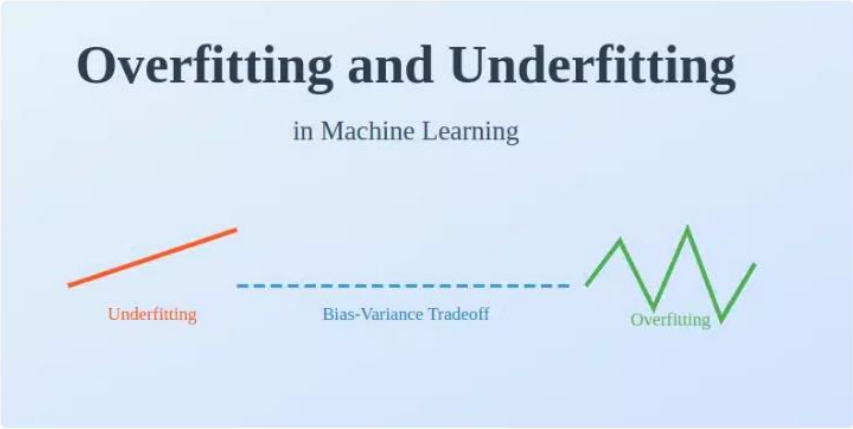
Aumentar os dados de treinamento.

Parar o treinamento antecipadamente para evitar memorização (também relacionado a épocas).

Trade-off entre viés e variância

- O trade-off viés/variância representa a relação entre a complexidade de um modelo e sua capacidade de generalização.

- Um modelo bem **balanceado** deve atingir um equilíbrio ideal entre viés e variância, garantindo que ele capture os padrões necessários sem memorizar ruídos.



Alto viés (underfitting): o modelo faz suposições fortes sobre os dados e não consegue capturar padrões.

Alta variância (overfitting): o modelo é muito sensível a pequenas flutuações nos dados, levando a uma generalização ruim.

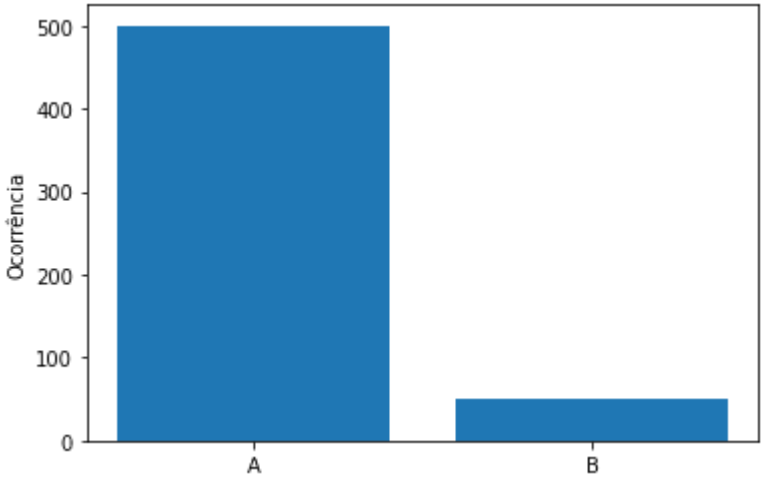
https://www.mygreatlearning.com/blog/overfitting-and-underfitting-in-machine-learning/

Desbalanceamento de Dados

Desbalanceamento dos Dados

- Ocorre quando há uma diferença significativa entre amostras das classes de um conjunto de dados relacionados à target.

- Por exemplo, imagine que na imagem abaixo A – casos em que não ocorreram fraude e B – casos em que ocorreram fraude.



Problemas do desbalanceamento

- As métricas de avaliações podem estar enviesadas **favorecendo a classe majoritária**, levando a conclusões equivocadas.

- Alguns algoritmos **necessitam** que as classes estejam balanceadas.

- Ocorre um impacto de ruído nas classes minoritárias, uma vez que os algoritmos de ML **tendem a tratar amostras pequenas como ruídos**, podendo até **descarta-las** durante o treinamento.

- Portanto, quanto mais dados de treinamento estiverem disponíveis, menos sensíveis serão os classificadores em relação às diferenças entre as classes.

Como resolver o problema de desbalanceamento?

Podemos usar duas abordagens:

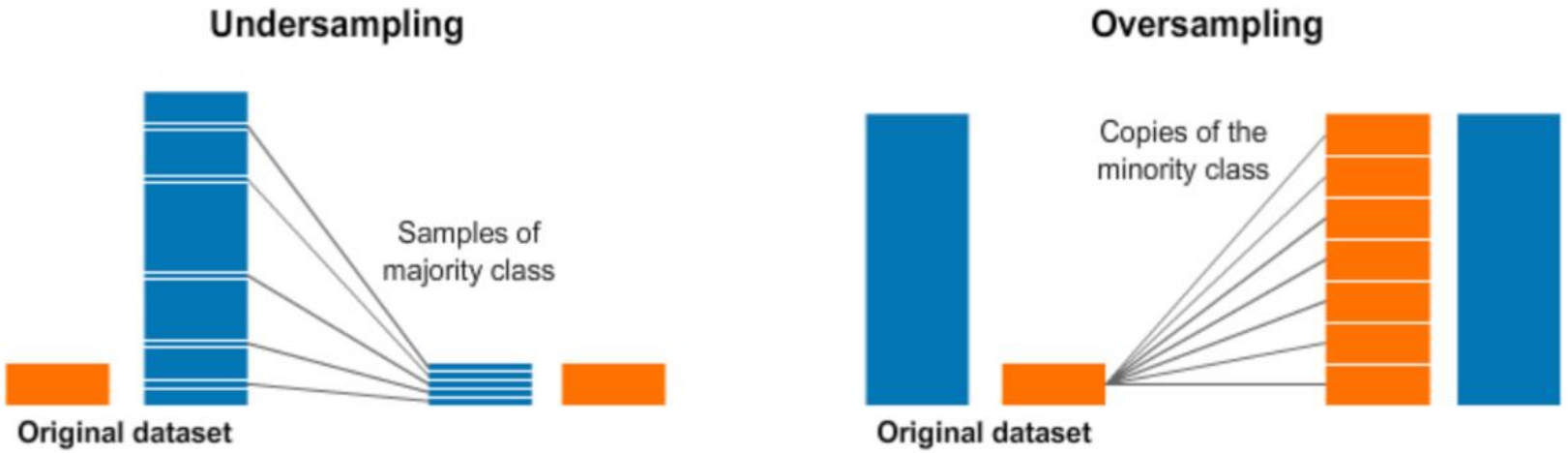
Sampling

É uma técnica para minimizar a discrepância entre as classes modificando as **distribuições** de cada uma no conjunto de treinamento.

Class Weight

É uma técnica para resolver o problema atribuindo **pesos maiores** às classes menos frequentes e **pesos menores** para as classes mais frequentes.

Undersampling x Oversampling



Class Weight

- É uma das estratégias mais “limpas” para resolver o problema de desbalanceamento de classes.
- Ao contrário das técnicas de amostragem (que manipulam o conjunto de dados), o Class Weight atua diretamente no algoritmo de aprendizado de máquina, influenciando como o modelo interpreta os erros durante o treinamento.
- Essa técnica é eficaz por focar em ajustar a Função de Custo (Loss Function) do modelo com maior **penalização para a classe minoritária.**

Class Weight força o modelo a:

Dar Mais Atenção: O algoritmo de aprendizado (como a Regressão Logística, Random Forest ou Redes Neurais) é obrigado a dar muito mais atenção aos padrões da classe minoritária.

Reduzir o Viés: O modelo não pode mais simplesmente prever a classe majoritária sempre para ter uma "boa" acurácia geral, pois o alto custo de errar a minoritária fará com que o custo total (loss) permaneça alto.

Aprender a Generalizar: Isso o incentiva a encontrar os limites de decisão que separam corretamente a classe rara, melhorando métricas críticas como **Recall** (sensibilidade) e **F1-Score** para essa classe.

→ Assuntos abordados nas próximas aulas...

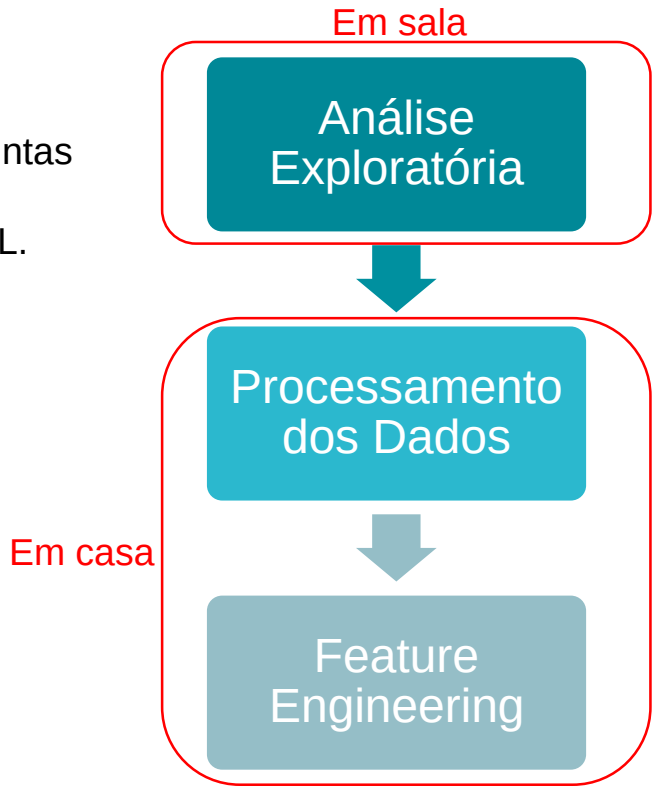
Tarefa

Tarefa

- Vocês receberão um dataset e um notebook contendo perguntas e direcionamentos para realizar toda a preparação e entendimento dos dados antes de rodarem um modelo de ML.

Importante!

1. Raciocínios e insights relevantes em formato de comentário serão apreciados na avaliação do exercícios;
2. Explicações de tomadas de decisões são obrigatórias.
3. A organização e limpeza do código serão avaliadas.




Obrigada! Nos vemos na próxima aula ☺


Facens
AQUI TEM ENGENHARIA