



# TRABALHO FINAL CD/IA

Caroline Bagatini, Carlo Laitano, Gabriella  
Parmeggianni, Gustavo Losch, Henrique Mayer,  
Henrique Kops



# ÍNDICE

- Introdução ----- 03
- Objetivos ----- 04
- Dataset ----- 05 - 06
- Pré-Processamento ----- 07 - 12
- Análise dos Dados ----- 13
- Modelagem ----- 14 - 15
- Avaliação ----- 16 - 18
- Conclusão ----- 19



# INTRODUÇÃO

Ciência de Dados é uma área que envolve a coleta, análise e interpretação de dados para obter insights valiosos e auxiliar na tomada de decisão. Neste trabalho, exploraremos um fluxo completo de análise e modelagem, seguindo as etapas essenciais do processo.

---

Ao longo do trabalho, nosso grupo enfrentou desafios e tomou decisões. Comentaremos sobre a escolha do dataset, dificuldades encontradas, mudanças realizadas e qualquer aspecto que tenha chamado nossa atenção durante a análise.

# OBJETIVOS



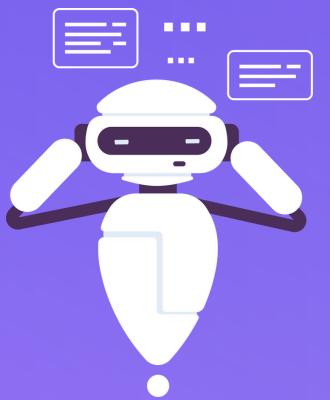
## OBJETIVO 01

Utilizar um conjunto de dados que permita uma **aplicação prática** e relevante do estudo.



## OBJETIVO 02

Desenvolver um modelo de aprendizado de máquina capaz de **prever se um jogo será premiado** com base nos atributos presentes no conjunto de dados.



## OBJETIVO 03

Possibilitar a aplicação prática dos resultados para a **tomada de decisões em cenários reais**.

# STEAM STORE DATASET

## Sobre o Dataset

O dataset escolhido aborda dados relevantes sobre diversos jogos vendidos na plataforma Steam. O conjunto original conta com **42497 instâncias e 24 atributos** e foi atualizado em 19 de maio de 2024.

## Atributos

Características importantes como as **informações gerais do jogo e avaliações dos usuários** são abordados detalhadamente no conjunto de dados e foram essenciais para a garantir boas métricas de avaliação ao final do treinamento.

# STEAM STORE DATASET

## Atributos

- **Binário Assimétrico:** age\_rating, linux/windows/mac support, categories e genres;
- **Racional Discreto:** dlc\_available;
- **Racional Contínuo:** overall\_review\_%, overall\_review\_count;
- **Ordinal:** overall\_review;

## Rótulo

A **variável alvo escolhida foi “awards”**, por meio dela, podemos saber se o jogo recebeu alguma premiação. O processo de pré processamento dos dados presentes no conjunto de dados original foi crucial para a obter conclusões significativas e melhorar o desempenho do modelo na predição do atributo rótulo.



# PRÉ-PROCESSAMENTO



**Código em Python**  
Split e OneHot Encoding dos atributos gênero e categoria por meio de código em python.

**Seleção dos Atributos Importantes**  
Retirada dos atributos que não seriam importantes para o treinamento do modelo.

**Seleção das Linhas**  
Retirada das linhas em que os valores eram nulos para os atributos gênero e categoria

**Transformação do Atributo “awards”**  
Discretização do atributo “awards” para “possui” ou “não possui” prêmios.

# 01 PRÉ-PROCESSAMENTO

Código em Python

```
# -*- coding: utf-8 -*-
from copy import copy
import pandas as pd
import numpy as np
from itertools import chain

#carregando arquivo
df = pd.read_csv('steam-games.csv')
df.head()
print(f"columns: {len(df.columns)}")
print(f"rows: {len(df)}")

#ajustando valores nulos
df['genres'] = df['genres'].fillna('')
df['categories'] = df['categories'].fillna('')
df = df[df['genres'].notnull() & df['categories'].notnull()]

#criando as colunas para categorias e generos
df['categories'] = df['categories'].apply(lambda x: x.replace(',', ' ', ',').split(','))
df['genres'] = df['genres'].apply(lambda x: x.replace(',', ' ', ',').split(','))
category_columns = set(chain.from_iterable(list(df['categories'].values)))
print(f"number of categories: {len(category_columns)}")
genre_columns = set(chain.from_iterable(list(df['genres'].values)))
print(f"number of genres: {len(genre_columns)}")
category_dict = { cat: np.zeros(42497) for cat in category_columns }
genre_dict = { cat: np.zeros(42497) for cat in genre_columns }

#atribuindo 0 ou 1 para as instancias do dataset
for i, row in df.iterrows():
    for cat in row['categories']:
        if cat in category_dict:
            category_dict[cat][i] = 1.0
    for genre in row['genres']:
        if genre in genre_dict:
            genre_dict[genre][i] = 1.0

#oragizando a nomenclatura das colunas de saida
lower_category_dict = {k.lower().replace(' ', '_'): v for k,v in category_dict.items()}
lower_category_dict['null_category'] = lower_category_dict.pop('')
lower_genre_dict = {k.lower().replace(' ', '_'): v for k,v in genre_dict.items()}
lower_genre_dict['null_genre'] = lower_genre_dict.pop('')

#criando colunas de genero e categoria
category_df = pd.DataFrame(lower_category_dict)
genre_df = pd.DataFrame(lower_genre_dict)
category_genre_df = pd.concat([category_df, genre_df], axis=1)

# validando se o join ocorreu com sucesso
category_genre_df[category_genre_df.isnull().any(axis=1)]

#resultado
result = pd.concat([df, category_genre_df], axis=1)

#validações
pd.set_option('display.max_columns', None)
len(result.columns)
new_cols = list(set(list(lower_category_dict.keys()) + list(lower_genre_dict.keys())))
new_cols.sort()
old_cols = list(set(result.columns) - set(new_cols))
old_cols.sort()
check_new_col_df = result[new_cols]
check_new_col_df[check_new_col_df.isnull().any(axis=1)]
result[['genres', 'categories']] + new_cols
result[old_cols + new_cols]

#salvando
result[old_cols + new_cols].to_csv('steam_games_preproc.csv', index=False)
```

# 01 PRÉ-PROCESSAMENTO

Código em Python



Antes

2 Colunas

68 Colunas



Depois

genres	categories
Action, Free to Play	Cross-Platform Multiplayer, Steam Trading ...
Action, Strategy, Free to Play	Steam Trading Cards, Steam Workshop, Stea...
Action, Adventure, Massively Multiplayer, Free ...	Online PvP, Stats, Remote Play on Phone, R...
Action, Indie, RPG, Early Access	Single-player, Captions available, Steam Clo...
Action, Adventure, Indie	Single-player, Online Co-op, Steam Cloud
Action, Adventure, Indie, Massively Multiplayer...	MMO, Online PvP, Online Co-op, Cross-Plat...
Indie, Simulation	Single-player, Online Co-op, Steam Achieve...
Simulation, Strategy, Early Access	Single-player, Steam Achievements, Steam ...
Indie, Simulation	Single-player, Online Co-op, Steam Achieve...
Action, Massively Multiplayer, Simulation, Free...	Single-player, MMO, Online PvP, Online Co...
Action	Single-player, Online PvP, Online Co-op, Ste...
Action, Adventure, Massively Multiplayer, Early...	MMO, Online PvP, Online Co-op, Family Sh...

action	nal_high-quality	adventure	nation_&_mode
1.0	0.0	0.0	0.0
1.0	0.0	0.0	0.0
1.0	0.0	1.0	0.0
1.0	0.0	0.0	0.0
1.0	0.0	1.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	1.0	0.0
1.0	0.0	1.0	0.0
1.0	0.0	1.0	0.0
1.0	0.0	0.0	0.0

# 02 PRÉ-PROCESSAMENTO

## Seleção dos Atributos Importantes



**Ignored (15)**

Filter	
S	categories
S	genres
S	about_description
S	content_descriptor
S	discount_percentage
S	discounted_price
S	original_price
S	release_date
S	title
N	recent_review_count
N	recent_review_%
C	recent_review
C	publisher
C	developer
N	app_id

Atributo	Missing
recent_review	36994(87%)

- Para selecionar os atributos que seriam utilizados no treinamento, algumas métricas foram utilizadas.
- Missing Values (recent\_review)
- Atributos Não Estruturados (description)
- Dimensionalidade (publisher e developer)
- Informações sem relação com o rótulo

# 03 PRÉ-PROCESSAMENTO

## Seleção das Linhas

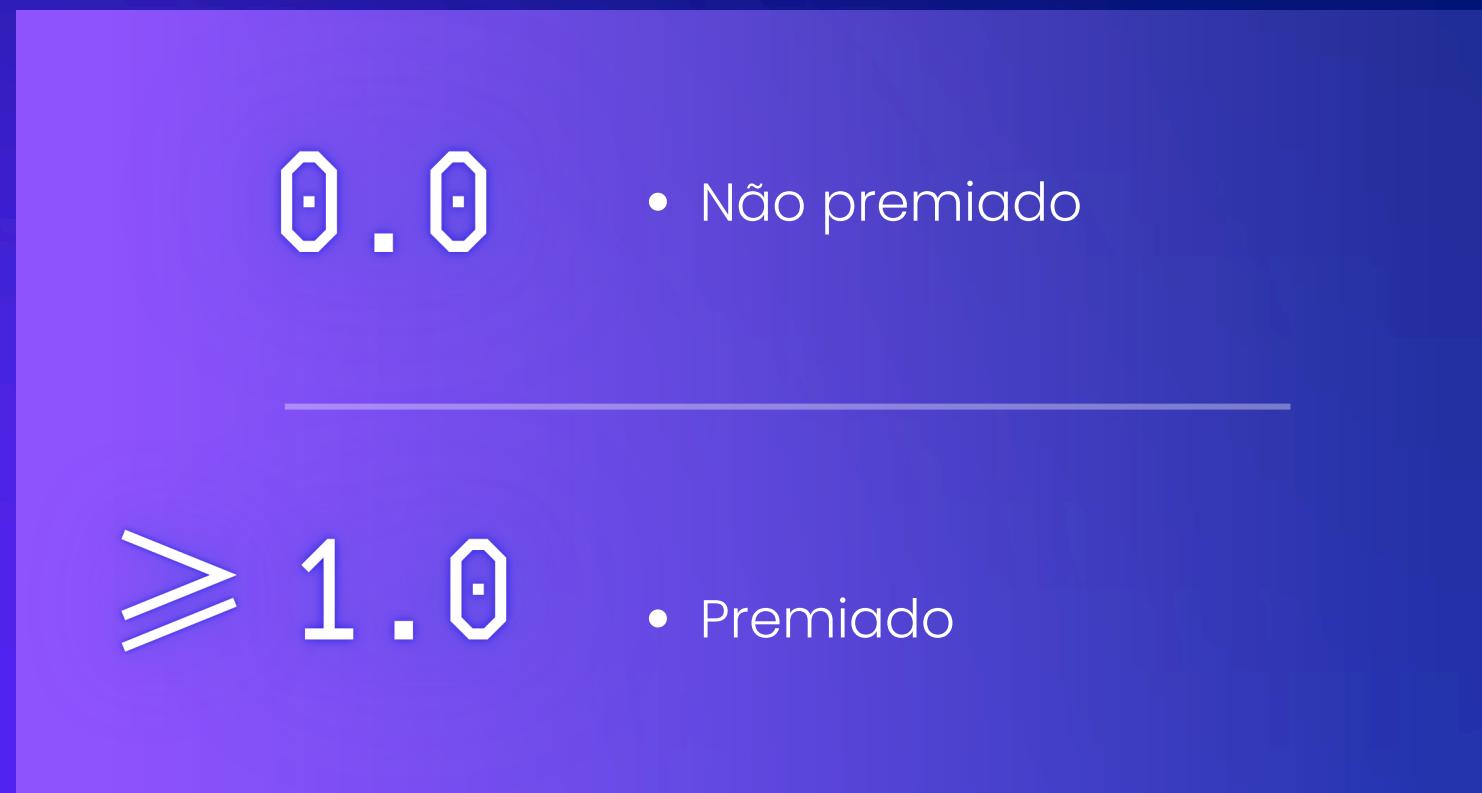
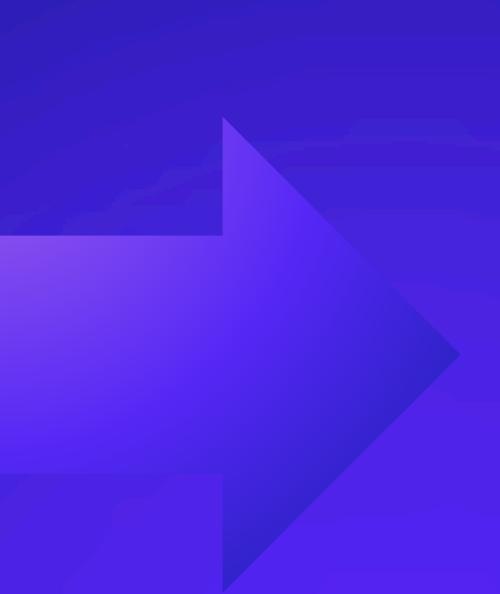
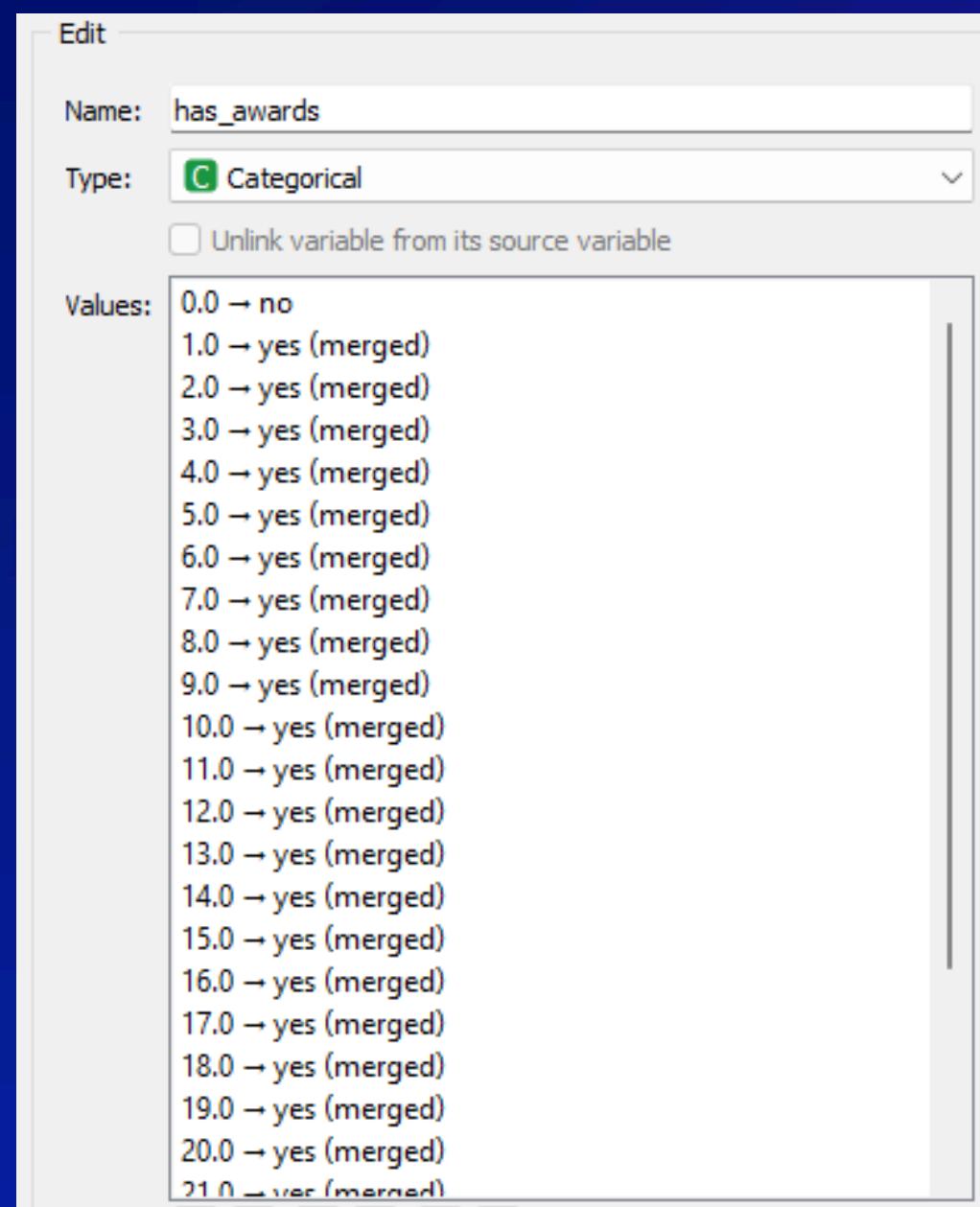
(03)

- Foi necessário retirar linhas com **valores nulos** em atributos importantes do dataset, como **"categories"** e **"genre"**.
- Dados nulos nesses atributos, mesmo que em pouca quantidade, poderiam prejudicar o aprendizado do modelo.

Atributo	Null
genre	87
categories	45

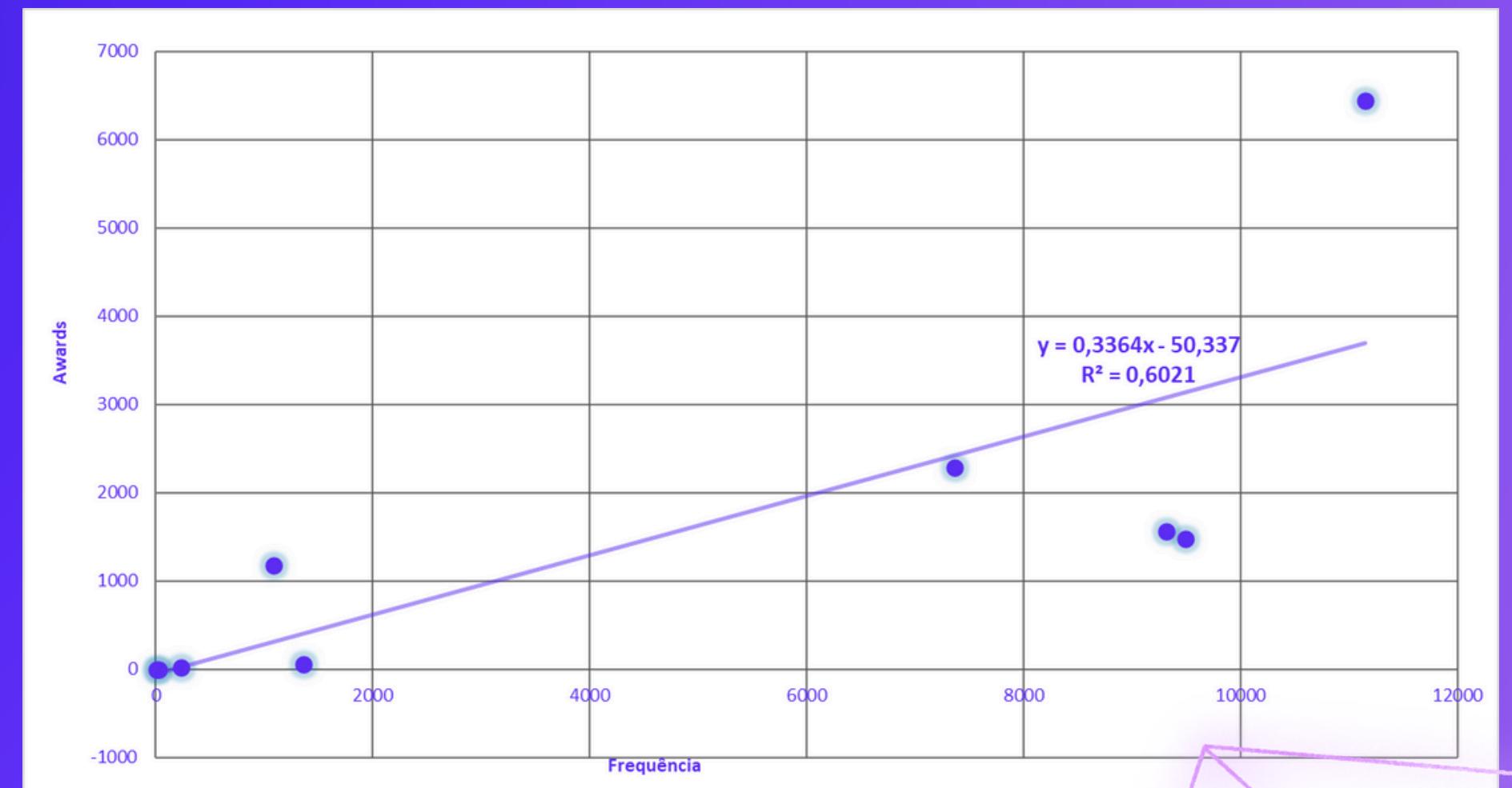
# 04 PRÉ-PROCESSAMENTO

Transformação do Atributo "awards"



# ANÁLISE DOS DADOS

Durante a análise, buscamos investigar a **correlação entre diversas variáveis**, portanto, identificando quais características contribuem para um maior número de prêmios. Um exemplo notável é a **relação entre as avaliações e a premiação**: observou-se que jogos com mais avaliações positivas tendem a receber mais prêmios, o que sugere uma forte correlação entre a participação do público e o reconhecimento por meio de premiações.



# MODELAGEM



# ESCOLHA DE MODELOS

03

- Foram considerados os modelos de aprendizado de máquina disponíveis no orange e que foram vistos em aula que retratam a **classificação binária**.
- Com isso, comparamos através de **cross-validation** com **k = 5 folds** os seguintes modelos:

**Random Forest**

**Tree**

**Neural Network**

**Naive Bayes**



# AVALIAÇÃO

Model	AUC	CA	F1 √	Prec	Recall	MCC
Random Forest	0.761	0.885	0.855	0.851	0.885	0.214
Neural Network	0.725	0.865	0.853	0.845	0.865	0.229
Tree	0.501	0.855	0.845	0.836	0.855	0.188
Naive Bayes	0.772	0.812	0.831	0.857	0.812	0.284

Modelo	Acurácia	F1	Precision	Recall
Random Forest	88%	85%	85%	88%
Neural Network	86%	85%	84%	86%

- **Precision** e **recall** são medidas cruciais para avaliar a performance de modelos de classificação.
- A **precision** mede a capacidade do modelo de identificar corretamente resultados positivos, enquanto o **recall** mede a capacidade de encontrar todos os resultados positivos reais.
- Não consideramos de forma significativa a **acurácia** pois o feature alvo é desbalanceada.
- Escolhemos avaliar os dois modelos que desempanharam melhor: **random forest** (1) e **neural network** (2)



# AVALIAÇÃO

## Modelo Random Forest

		Predicted		$\Sigma$
		no	yes	
Actual	no	36860	687	37547
	yes	4193	661	4854
$\Sigma$		41053	1348	42401

- Podemos perceber que o modelo gerado pela **random forest** tem um **viés** voltado para a **classe negativa**, pois o dataset possui menos instâncias da classe positiva (classe rara).

- Taxa de erro:  $( 687 + 4193 ) / 42401 = \sim 11\%$
- Taxa de falsos negativo:  $( 4193 / 4193 + 661 ) = \sim 86\%$

Random Forest	Métrica
Taxa de Erro	11%
Taxa de Falsos Negativos	86%

# AVALIAÇÃO

## Modelo Neural Network

		Predicted		$\Sigma$
		no	yes	
Actual	no	35500	2047	37547
	yes	3661	1193	4854
$\Sigma$		39161	3240	42401

- Taxa de erro:  $( 2047 + 3661 ) / 42401 = \sim 13\%$
- Taxa de falsos negativo:  $( 3661 / 3661 + 1193 ) = \sim 75\%$

- Já o modelo gerado pela **rede neural**, apesar de ter a F1-score afetada por uma taxa maior de erros (+3%), **diminuiu o viés** na classe negativa em ~11%.

Métricas	Neural Network	Random Forest
Taxa de Erro	13%	11%
Taxa de Falsos Negativos	75%	86%

# CONCLUSÃO

Model	AUC	CA	F1 √	Prec	Recall	MCC
Random Forest	0.761	0.885	0.855	0.851	0.885	0.214
Neural Network	0.725	0.865	0.853	0.845	0.865	0.229
Tree	0.501	0.855	0.845	0.836	0.855	0.188
Naive Bayes	0.772	0.812	0.831	0.857	0.812	0.284

- Com isso, escolhemos o modelo de rede neural pois o mesmo conseguiu distribuir melhor suas decisões dentre as classes **positiva** (ganhar prêmio - rara) e **negativa** (não ganhar prêmio - comum).

OBRIGADO!



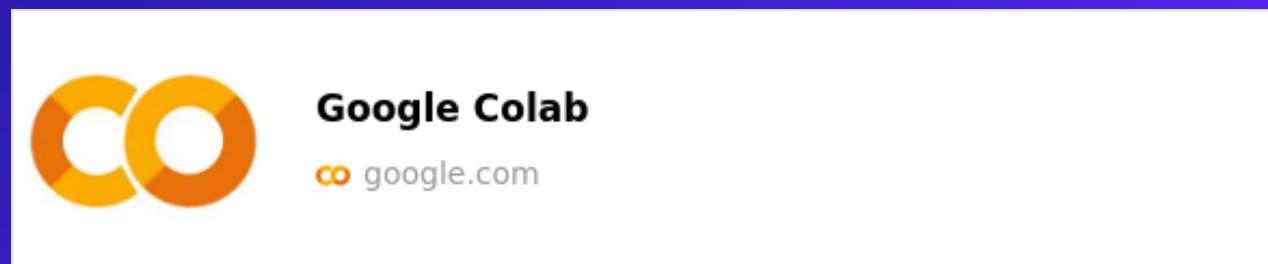
# REFERÊNCIAS



## Steam Store Data

The ultimate dataset for game developers and analysts.

[kaggle.com](https://www.kaggle.com)

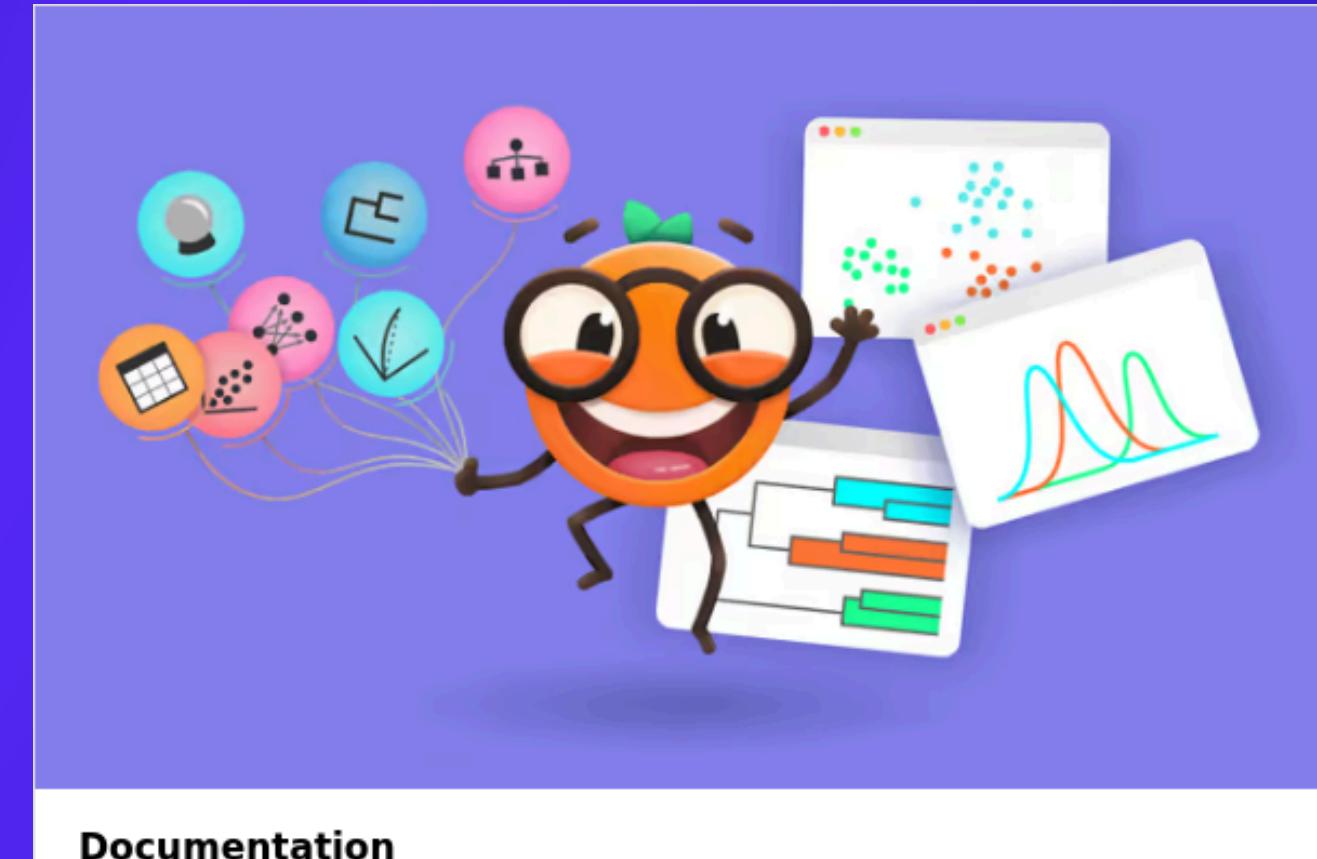


## Google Colab

[google.com](https://colab.research.google.com)



## Arquivos - TF Intro a CD



## Documentation

Orange Data Mining Toolbox

[Orange Data Mining /](https://orangedatamining.org/)

