```
rarity(metadata, padding = False, save_output = False, path = "./", name = "Collection", plotting =
    True, plt_xaxis = "Harmonic", plt_yaxis = "Geometric", ind_sig = 0.99, save_graph = False)
```

Dependencies

To use the `rarity` class, ensure you have the following Python packages installed: `os` (standard libraries), `pandas`, `numpy`, `scipy`, `matplotlib`.

If using the notebook version of this package, ensure your environment supports Jupyter or is run in a compatible IDE (e.g., VSCode, JupyterLab).

Inputs

Parameter	Type	Default	Description
metadata	`pd.DataFrame`	–	Metadata DataFrame of the NFT collection. Each row is a token with its ID as the index; each column is a trait with the column name as the trait name.
padding	`bool`	TRUE	Whether to pad missing trait values. If True, missing data is filled with a placeholder (e.g., 'None').
save_output	`bool`	False	Whether to save output data into CSVs.
path	`str`	'./'	Directory path where output data will be saved.
name	`str`	'Collection'	Name of the NFT collection. Used in output CSV filenames and plot titles.
plotting	`bool`	TRUE	Whether to generate visualizations comparing rarity ranks across methods.
plt_xaxis	`str`	'Geometric'	Name of the rarity method for the x-axis of the comparison plot (options include 'Harmonic', 'Geometric', 'Arithmetic', 'TN_Harmonic', 'TN_Geometric', 'TN_Arithmetic', where 'TN' stands for trait-normalized').
plt_yaxis	`str`	'Harmonic'	Name of the rarity method for the y-axis of the comparison plot (options include 'Harmonic', 'Geometric', 'Arithmetic', 'TN_Harmonic', 'TN_Geometric', 'TN_Arithmetic', where 'TN' stands for trait-normalized').
ind_sig	`float`	0.99	Threshold for trait independence test. Passing the test indicates that the pair of traits tested are independent.
save_graph	`bool`	False	If True, scatter plots are exported as a standalone png file.

Methods

```
rarity.run()
```

Executes the full rarity analysis pipeline:

- Computes trait frequency matrix
- Calculates multiple rarity values, scores and ranks such as 'Harmonic', 'Geometric', 'Arithmetic', 'TN_Harmonic', 'TN_Geometric', 'TN_Arithmetic'.
- Assesses trait independence
- Optionally save these outputs into CSVs
- Optionally generates visual plots for rarity comparison and save plots

Attributes

Attribute	Description
original_padding	A string explaining whether the metadata has already been padded prior to download.
metadata	DataFrame of metadata after padding and cleaning.
attr_prob	DataFrame containing the token-level trait frequencies.
rarity_metrics	DataFrame containing rarity values, scores and ranks across all methods for each token.
independ	Matrix showing pairwise trait independence evaluations and criteria.
ind_pair_result	A dictionary containing number of independent trait pair (ITP), number of total trait pairs (TTP), and ITP/TTP.
cramersV	DataFrame containing Cramer's V test result.

Output

- save_output = True exports attr_prob, rarity_metrics, independ, cramersV.
- save_graph = True (plotting = True required) exports a rank–rank comparison plot between methods.

Example

```
import pandas as pd
import os
import matplotlib.pyplot as plt
import numpy as np
import scipy.stats as stats
from scipy.stats.distributions import chi2

from Func_rarity import rarity # if using Func_rarity.py, or otherwise %run Func_rarity.ipynb

# Load metadata
path = os.getcwd()
collection_name = "BoredApeYachtClub"
Address = "0xBC4CA0EdA7647A8aB7C2061c2E118A18a936f13D"
metadata = pd.read_csv(collection_name + "_Address_" + Address + "_metadata.csv")

# Create rarity analyzer
rc = rarity(
    metadata = metadata,
    padding = False,
    save_output = False,
    path = path,
    name = collection_name,
    plotting = True,
    plt_xaxis = "Harmonic",
    plt_yaxis = "Geometric",
    ind_sig = 0.99,
    save_graph = True
)

# Run rarity analysis
rc.run()

# Access attributes such as rarity metrics
rc.rarity_metrics
```

Notes

- This class assumes a tabular structure: rows = tokens, columns = traits.
- Missing traits are optionally padded to ensure fair scoring.
- Trait independence is based on statistical similarity of value distributions.