```
%pip install discopula
```

```
Collecting discopula
  Downloading discopula-0.2.1-py3-none-any.whl.metadata (5.2 kB)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from discopula) (1.26.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from discopula) (1.13.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from discopula) (3.8.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula) (0.1
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula) (
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula) (11
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopula)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->discopu
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotli
Downloading discopula-0.2.1-py3-none-any.whl (39 kB)
Installing collected packages: discopula
Successfully installed discopula-0.2.1
```

> Make sure to have discopula's latest version installed using `pip`. More information about the latest version can be found at https://pypi.org/project/discopula/

```
import numpy as np
from discopula import GenericCheckerboardCopula
```

Create Sample Contingency Table and Initialize the GenericCheckerboardCopula

```
contingency_table = np.array([
    [0, 0, 20],
    [0, 10, 0],
    [20, 0, 0],
    [0, 10, 0],
    [0, 0, 20]
])
copula = GenericCheckerboardCopula.from_contingency_table(contingency_table)
print(f"Shape: {copula.P.shape}")
print(f"Probability matrix P:\n{copula.P}")

# Getting Back the contingency table mid-way at any given time
reconstructed_table = copula.contingency_table
print(reconstructed_table)
```

```
Shape: (5, 3)
Probability matrix P:
[[0.    0.    0.25 ]
 [0.    0.125 0.   ]
 [0.25  0.    0.   ]
 [0.    0.125 0.   ]
 [0.    0.    0.25 ]]
[[0 0 2]
 [0 1 0]
 [2 0 0]
 [0 1 0]
 [0 0 2]]
```

Calculating CCRAM & SCCRAM (non-vectorized)

```
ccram_0_to_1 = copula.calculate_CCRAM(from_axis=0, to_axis=1)
ccram_1_to_0 = copula.calculate_CCRAM(from_axis=1, to_axis=0)
print(f"CCRAM 0->1: {ccram_0_to_1:.4f}")
print(f"CCRAM 1->0: {ccram_1_to_0:.4f}")

sccram_0_to_1 = copula.calculate_CCRAM(from_axis=0, to_axis=1, is_scaled=True)
sccram_1_to_0 = copula.calculate_CCRAM(from_axis=1, to_axis=0, is_scaled=True)
print(f"SCCRAM 0->1: {sccram_0_to_1:.4f}")
print(f"SCCRAM 1->0: {sccram_1_to_0:.4f}")
```

```
CCRAM 0->1: 0.8438
CCRAM 1->0: 0.0000
SCCRAM 0->1: 1.0000
SCCRAM 1->0: 0.0000
```

Calculating CCRAM & SCCRAM (vectorized)

```
ccram_0_to_1_vec = copula.calculate_CCRAM_vectorized(from_axis=0, to_axis=1)
ccram_1_to_0_vec = copula.calculate_CCRAM_vectorized(from_axis=1, to_axis=0)
print(f"CCRAM 0->1: {ccram_0_to_1_vec:.4f}")
print(f"CCRAM 1->0: {ccram_1_to_0_vec:.4f}")

sccram_0_to_1_vec = copula.calculate_CCRAM_vectorized(from_axis=0, to_axis=1, is_scaled=True)
sccram_1_to_0_vec = copula.calculate_CCRAM_vectorized(from_axis=1, to_axis=0, is_scaled=True)
print(f"SCCRAM 0->1: {sccram_0_to_1_vec:.4f}")
print(f"SCCRAM 1->0: {sccram_1_to_0_vec:.4f}")
```

```
CCRAM 0->1: 0.8438
CCRAM 1->0: 0.0000
SCCRAM 0->1: 1.0000
SCCRAM 1->0: 0.0000
```

Getting Category Predictions

```
predictions_0_to_1 = copula.get_category_predictions(0, 1)
print("\nPredictions from axis 0 to axis 1:")
print(predictions_0_to_1)
predictions_1_to_0 = copula.get_category_predictions(1, 0, "Y", "X")
print("\nPredictions from axis 1 to axis 0:")
print(predictions_1_to_0)
```

```
Category Predictions: X → Y
----------------------------------------

Predictions from axis 0 to axis 1:
   X Category  Predicted Y Category
0           0                     2
1           1                     1
2           2                     0
3           3                     1
4           4                     2

Category Predictions: Y → X
----------------------------------------

Predictions from axis 1 to axis 0:
   Y Category  Predicted X Category
0           0                     2
1           1                     2
2           2                     2
```