```
%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/

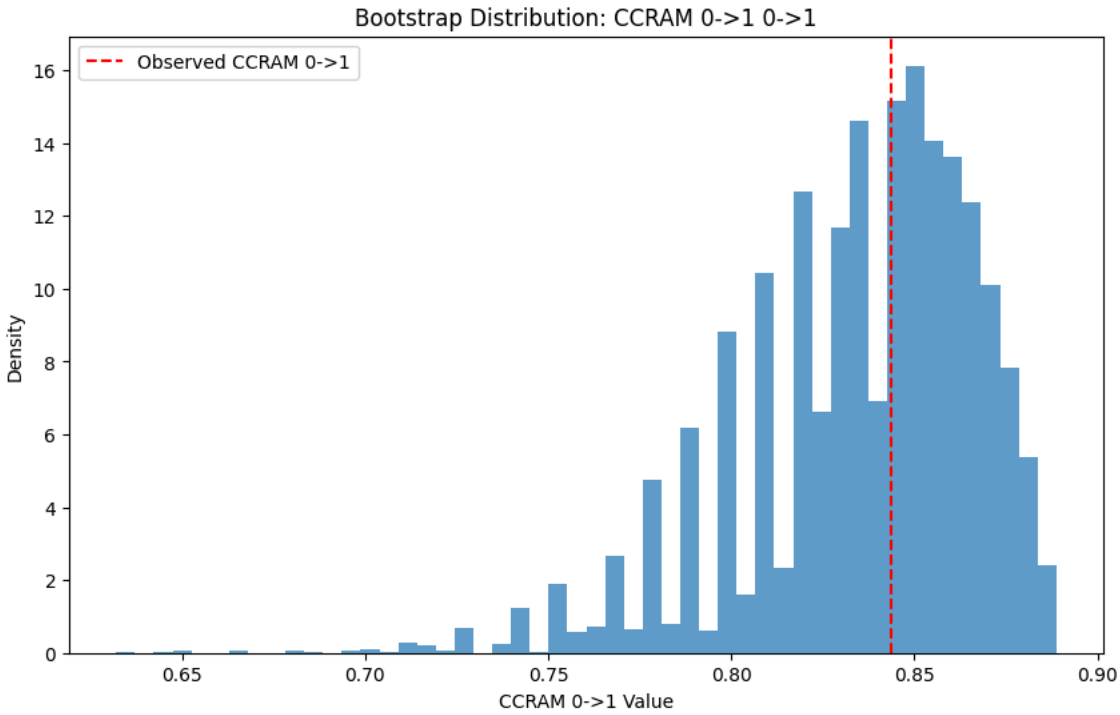
```
import numpy as np
from discopula import (
    bootstrap_ccram,
    permutation_test_ccram,
    bootstrap_predict_category_summary,
    display_prediction_summary
)
```

Create Sample Contingency Table

```
contingency_table = np.array([
    [0, 0, 20],
    [0, 10, 0],
    [20, 0, 0],
    [0, 10, 0],
    [0, 0, 20]
])
```

Bootstrapping CCRAM & SCCRAM Metrics

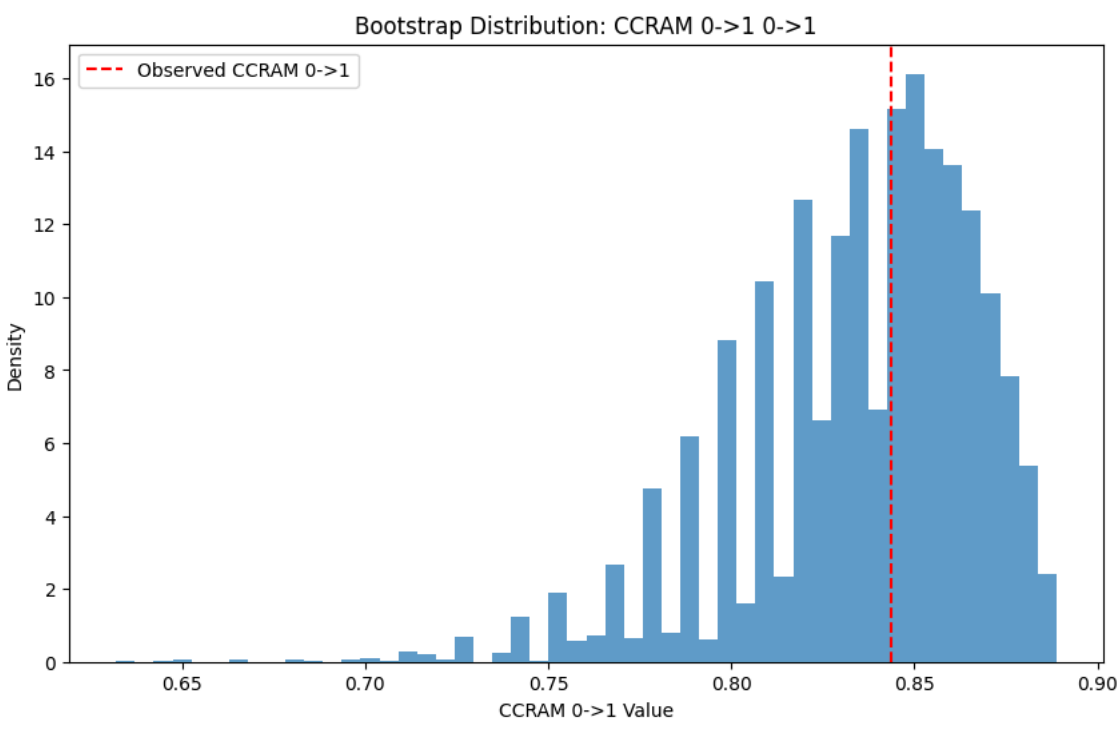
```
ccram_result = bootstrap_ccram(
    contingency_table,
    from_axis=0,
    to_axis=1,
    n_resamples=9999
)
```




```
print(f"Metric Name: {ccram_result.metric_name}")
print(f"Observed Value: {ccram_result.observed_value:.4f}")
print(f"95% CI: ({ccram_result.confidence_interval[0]:.4f}, {ccram_result.confidence_interval[1]:.4f})")
print(f"Standard Error: {ccram_result.standard_error:.4f}")
print(f"Bootstrap Distribution: {type(ccram_result.bootstrap_distribution)}")
```

```
Metric Name: CCRAM 0->1
Observed Value: 0.8438
95% CI: (0.7553, 0.8809)
Standard Error: 0.0325
Bootstrap Distribution: <class 'numpy.ndarray'>
```

```
ccram_result.histogram_fig
```



Bootstrap Prediction of Categories through Checkerboard Copula Regression

```
prediction_matrix = bootstrap_predict_category_summary(
    contingency_table,
    from_axis=0,
```

```
    to_axis=1,
    n_resamples=9999
)
print("\nPrediction Matrix:")
print(prediction_matrix)
```

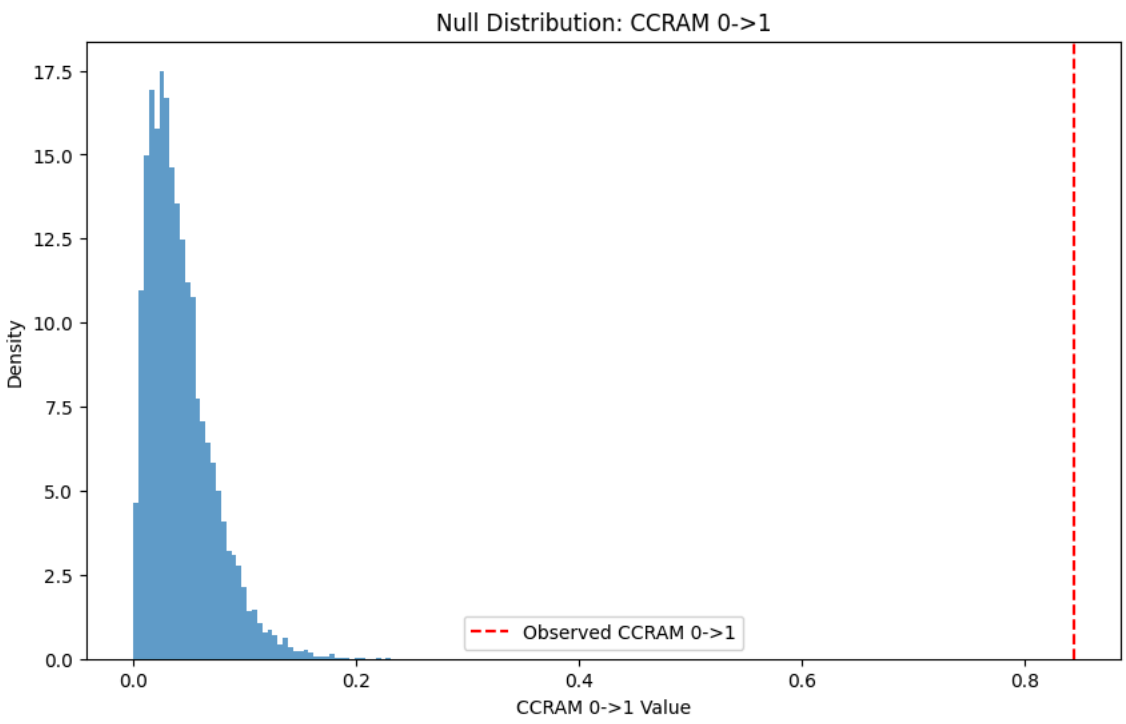
```
Prediction Matrix:
[[0.00000000e+00 0.00000000e+00 1.00000000e+02 1.00010001e-02
  0.00000000e+00]
 [0.00000000e+00 1.00000000e+02 0.00000000e+00 9.99899990e+01
  0.00000000e+00]
 [1.00000000e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00
  1.00000000e+02]]
```

```
display_prediction_summary(
    prediction_matrix,
    from_axis_name="X1",
    to_axis_name="X2"
)
```

```
Prediction Summary (% of bootstrap samples)
From X1 to X2:
--------------------------------------------------
       X1=0   X1=1   X1=2   X1=3   X1=4
X2=0   0.0%   0.0% 100.0%   0.0%   0.0%
X2=1   0.0% 100.0%   0.0% 100.0%   0.0%
X2=2 100.0%   0.0%   0.0%   0.0% 100.0%
--------------------------------------------------
```

Permutation Testing for CCRAM & SCCRAM Metrics

```
perm_result = permutation_test_ccram(
    contingency_table,
    from_axis=0,
    to_axis=1,
    alternative='greater',
    n_resamples=9999
)
```




```
print(f"Metric Name: {perm_result.metric_name}")
print(f"Observed Value: {perm_result.observed_value:.4f}")
print(f"P-Value: {perm_result.p_value:.4f}")
print(f"Null Distribution: {type(perm_result.null_distribution)}")
```

```
Metric Name: CCRAM 0->1
Observed Value: 0.8438
P-Value: 0.0001
Null Distribution: <class 'numpy.ndarray'>
```

```
perm_result.histogram_fig
```