Machine Learning Lab Assignment 2

Shreeyash S. Dongarkar

22510025 ( T2 )

Normalization Methods

Code:

*import* streamlit *as* st  
*import* numpy *as* np  
*import* scipy.stats *as* stats  
*import* pandas *as* pd  
*import* seaborn *as* sns  
*import* matplotlib.pyplot *as* plt  
*from* sklearn.preprocessing *import* quantile\_transform  
  
*# Sidebar Inputs*st.sidebar.header("Variable Settings")  
size = st.sidebar.number\_input("Size of Data", value=10000)  
mean\_B = st.sidebar.number\_input("Mean of B (Gaussian)", value=5.0)  
sd\_B = st.sidebar.number\_input("Standard Deviation of B (Gaussian)", value=2.0)  
a\_I = st.sidebar.number\_input("Shape Parameter of I (Power Law)", value=0.3)  
p\_H = st.sidebar.number\_input("Probability of H (Geometric)", value=0.005)  
quantile\_output = st.sidebar.radio(  
 "Select Output Distribution for Quantile Transform", ["normal", "uniform"], index=0  
)  
  
*# Generate Data*np.random.seed(42)  
B = np.random.normal(mean\_B, sd\_B, size)  
I = stats.powerlaw.rvs(a=a\_I, size=size)  
H = stats.geom.rvs(p=p\_H, size=size)  
  
data = pd.DataFrame({"B": B, "I": I, "H": H})  
  
  
*# Normalization Functions  
def* normalize(data, method):  
 df = data.copy()  
 *if* method == "Divide by Max":  
 df = df / df.max()  
 *elif* method == "Divide by Sum":  
 df = df / df.sum()  
 *elif* method == "Z-score":  
 df = df.apply(stats.zscore)  
 *elif* method == "Percentile Scaling":  
 df = df.rank(pct=*True*)  
 *elif* method == "Median Normalization":  
 medians = df.median()  
 target\_median = medians.mean()  
 multipliers = target\_median / medians  
 df \*= multipliers  
 *elif* method == "Quantile Normalization":  
 df = pd.DataFrame(  
 quantile\_transform(df, output\_distribution=quantile\_output, copy=*True*),  
 columns=df.columns,  
 )  
 *return* df  
  
  
methods = [  
 "Divide by Max",  
 "Divide by Sum",  
 "Z-score",  
 "Percentile Scaling",  
 "Median Normalization",  
 "Quantile Normalization",  
]  
  
*for method in methods:  
 normalized\_data = normalize(data, method)  
  
 st.subheader(f"Normalization Method: {method}")  
  
 fig, axes = plt.subplots(2, 1, figsize=(14, 10), gridspec\_kw={"hspace": 0.3})  
 axes[0].boxplot(  
 [B, I, H],  
 labels=["B (Gaussian)", "I (Power Law)", "H (Geometric)"],  
 vert=False,  
 )  
 axes[0].set\_title("Before Normalization")  
 axes[0].set\_xlabel("Values")  
 axes[0].grid(axis="x")  
  
 axes[1].boxplot(  
 [normalized\_data["B"], normalized\_data["I"], normalized\_data["H"]],  
 labels=["B", "I", "H"],  
 vert=False,  
 )  
 axes[1].set\_title("After Normalization")  
 axes[1].set\_xlabel("Values")  
 axes[1].grid(axis="x")  
  
 st.pyplot(fig)  
  
 fig, axes = plt.subplots(3, 1, figsize=(14, 12), gridspec\_kw={"hspace": 0.4})  
 for idx, (name, original, normalized, xlabel, ylabel) in enumerate(  
 zip(  
 data.columns,  
 [B, I, H],  
 [normalized\_data["B"], normalized\_data["I"], normalized\_data["H"]],  
 [  
 "Value of B (Continuous)",  
 "Value of I (Continuous)",  
 "Number of Trials Until First Success",  
 ],  
 ["Frequency", "Frequency", "Count of Observations"],  
 )  
 ):  
 axes[idx].hist(  
 original, bins=50, alpha=0.5, label=f"{name}\_original", color="blue"  
 )  
 axes[idx].hist(  
 normalized, bins=50, alpha=0.5, label=f"{name}\_normalized", color="orange"  
 )  
 axes[idx].set\_title(  
 f"Comparison of Original and Normalized Versions for {name}"  
 )  
 axes[idx].set\_xlabel(xlabel)  
 axes[idx].set\_ylabel(ylabel)  
 axes[idx].legend()  
  
 st.pyplot(fig)*

Before applying Normalization Methods:

Gaussian Distribution

* This is a distribution with given mean and standard distribution where values cluster around mean with few extreme values on either side.
* It is bell-shaped curve.

Power Law Distribution

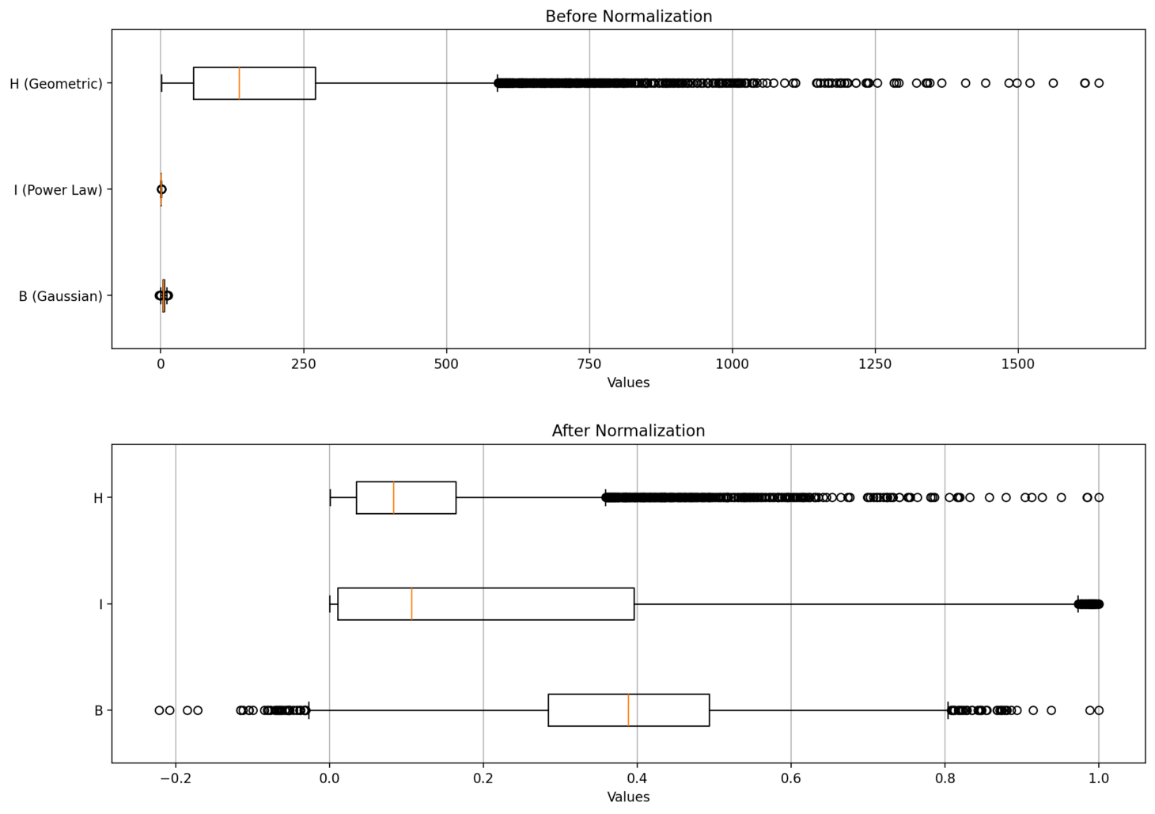
* For smaller value of ‘a’ it is right-skewed.
* Here smaller ‘a’ values lead to higher probability of generating large values leading to steeper curve
* As the value of ‘a’ increases probability of generating higher values decreases leading to less steep curve

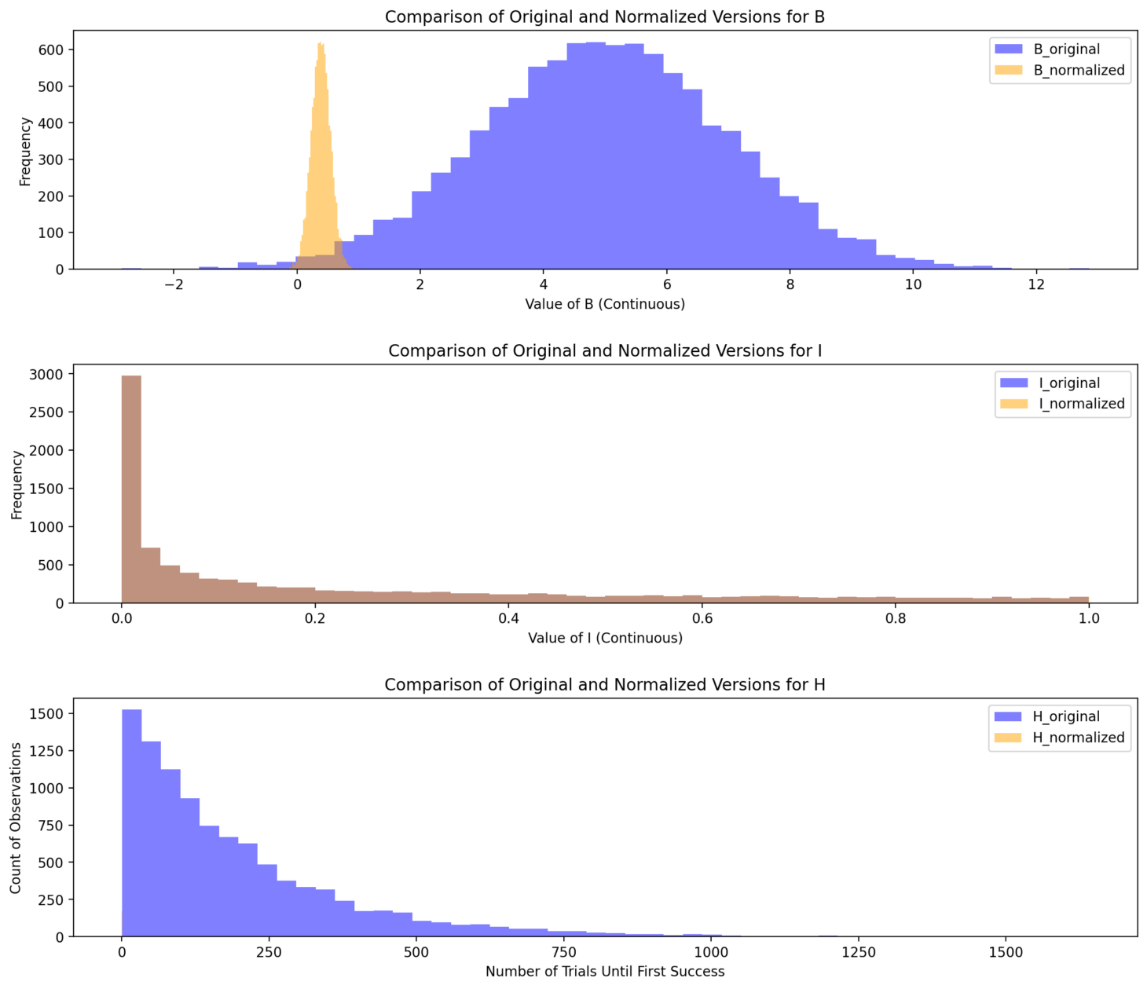
Geometric Distribution

* For smaller ‘p’ it leads to large number of failures leading to more concentration near lower end creating highly skewed curve

**NORMALIZATION METHODS:**

**Divide by Max:**





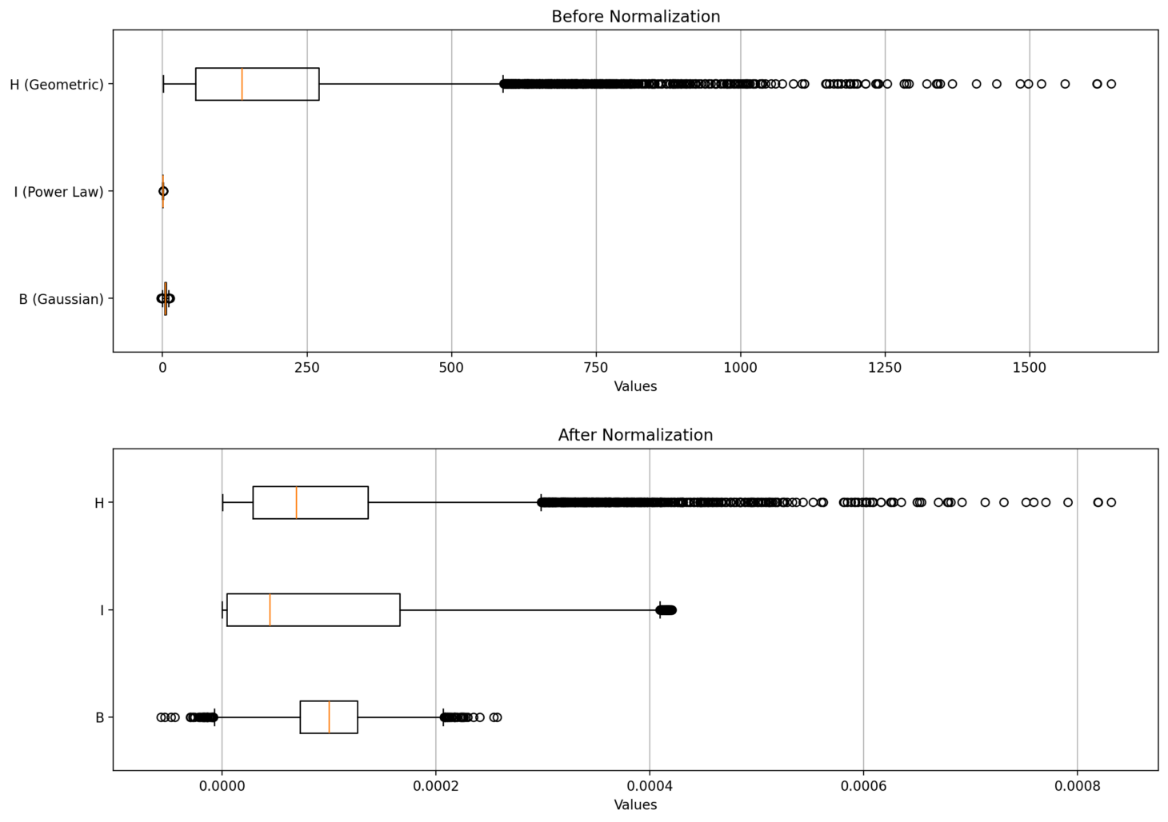
B: Shape still remains bell-like but dividing by max value compresses the entire distribution into range [0, 1] and if negative values are present then [-1, 1] can be the range.

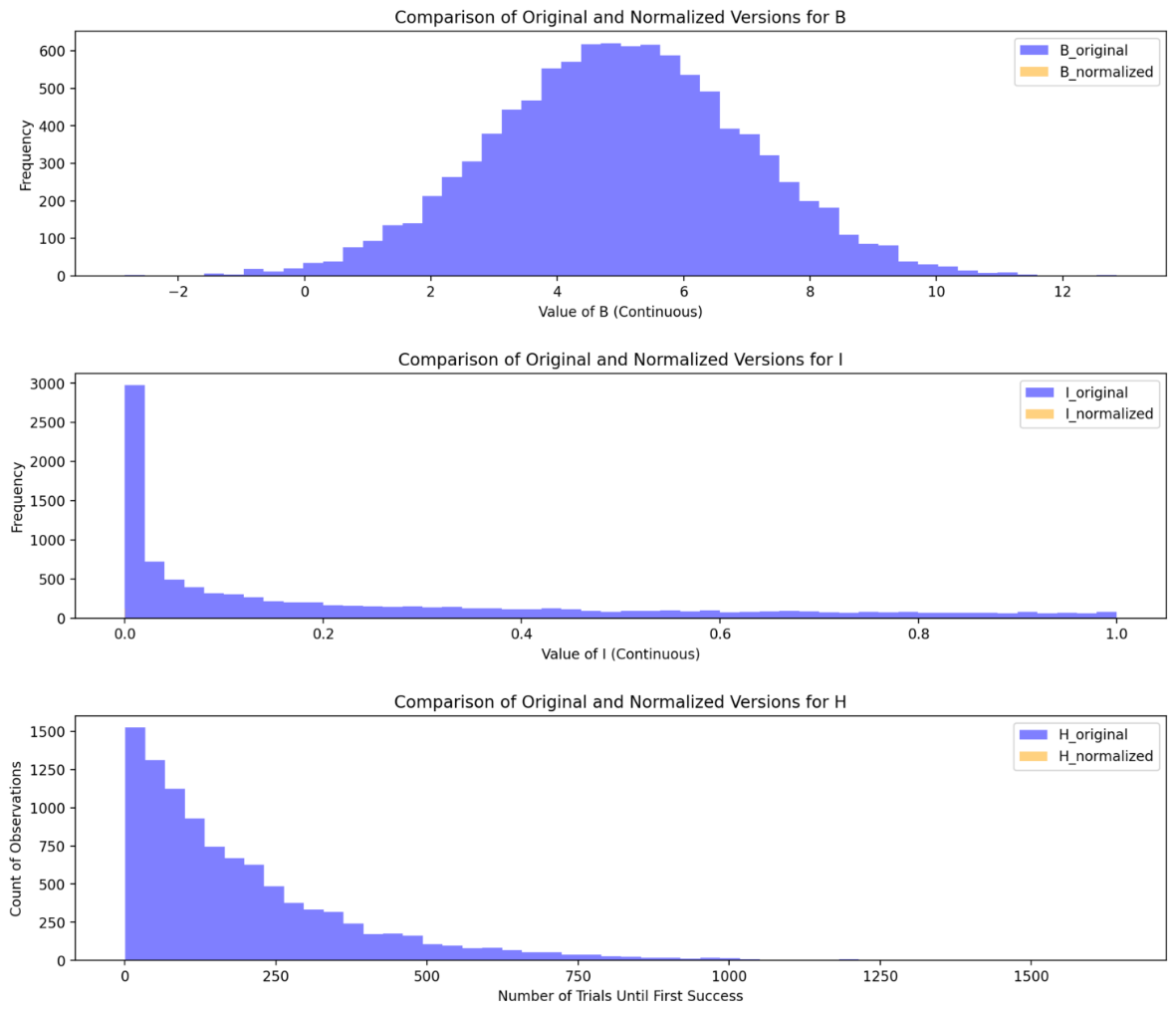
I: Shape remains the same with no effects as distribution is already in range [0, 1]

H: Shape remains the same but the small values are shrunk close to 0 but discrete nature is preserved.

* Is good for keeping original distribution intact while normalizing within [0, 1]
* Not suitable for handling skewed distributions

**Divide by Sum:**





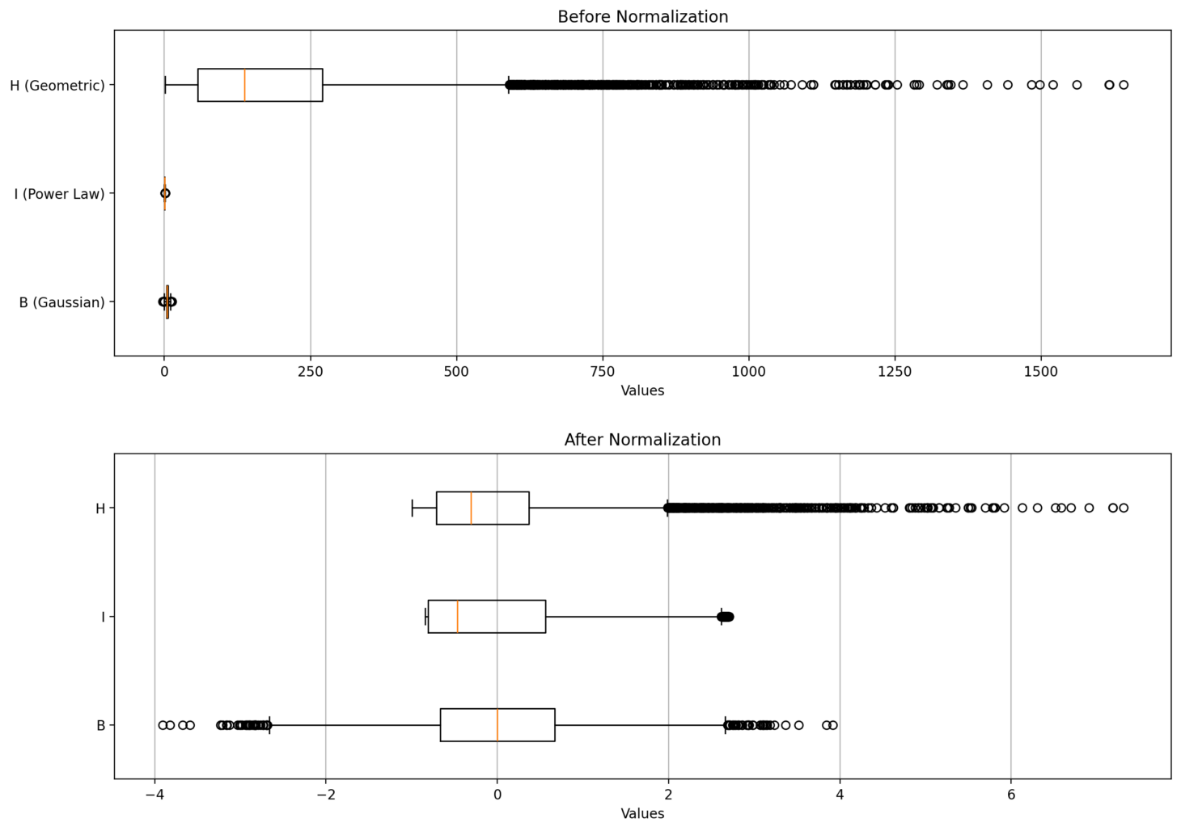
B: Heavily scales down values but shape remains intact.

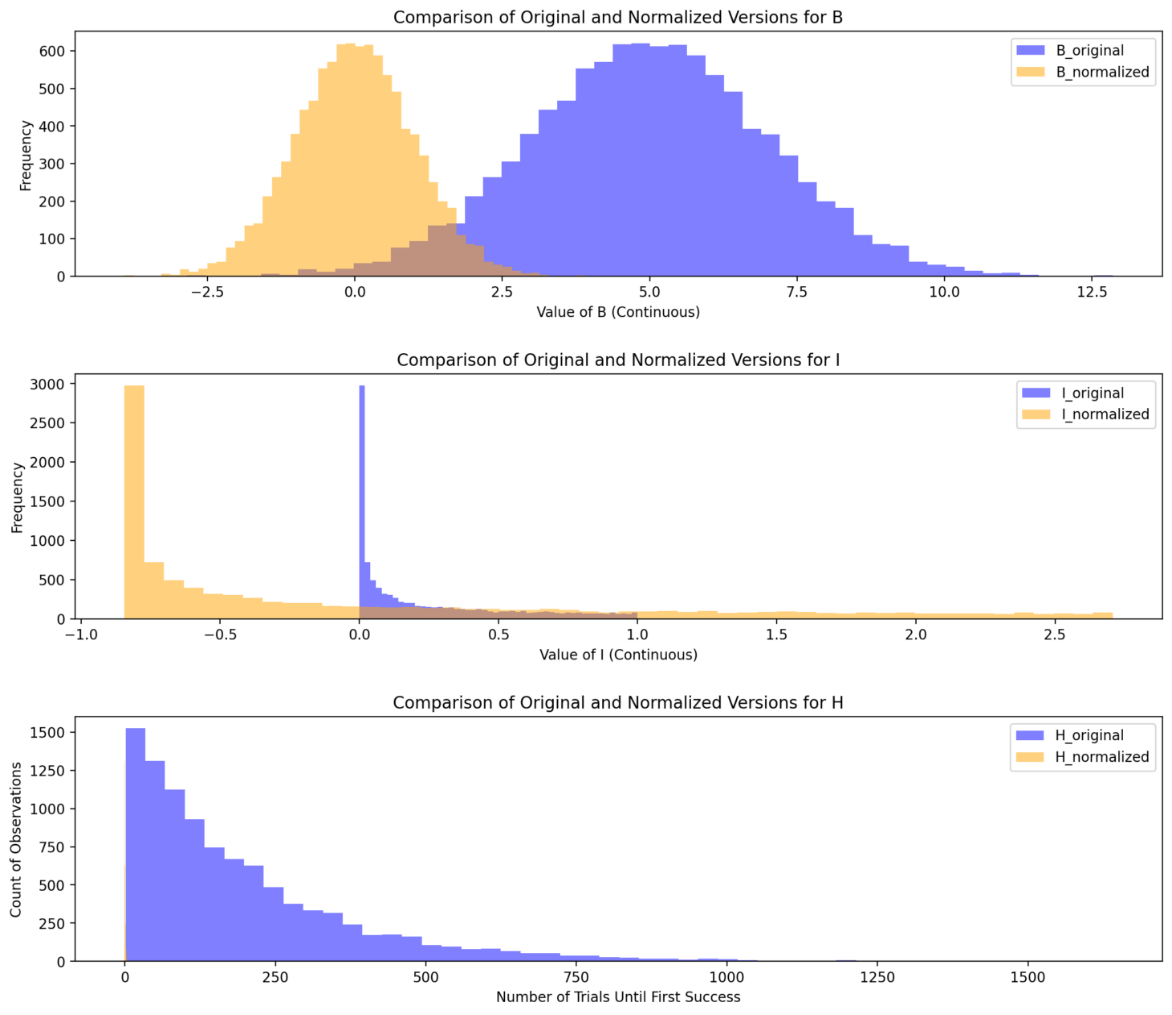
I: Distribution retains skewness but small values get compressed even further while large values remain relatively high.

H: Values are scaled down significantly.

* Not suitable for handling skewed distributions

**Z-Score:**



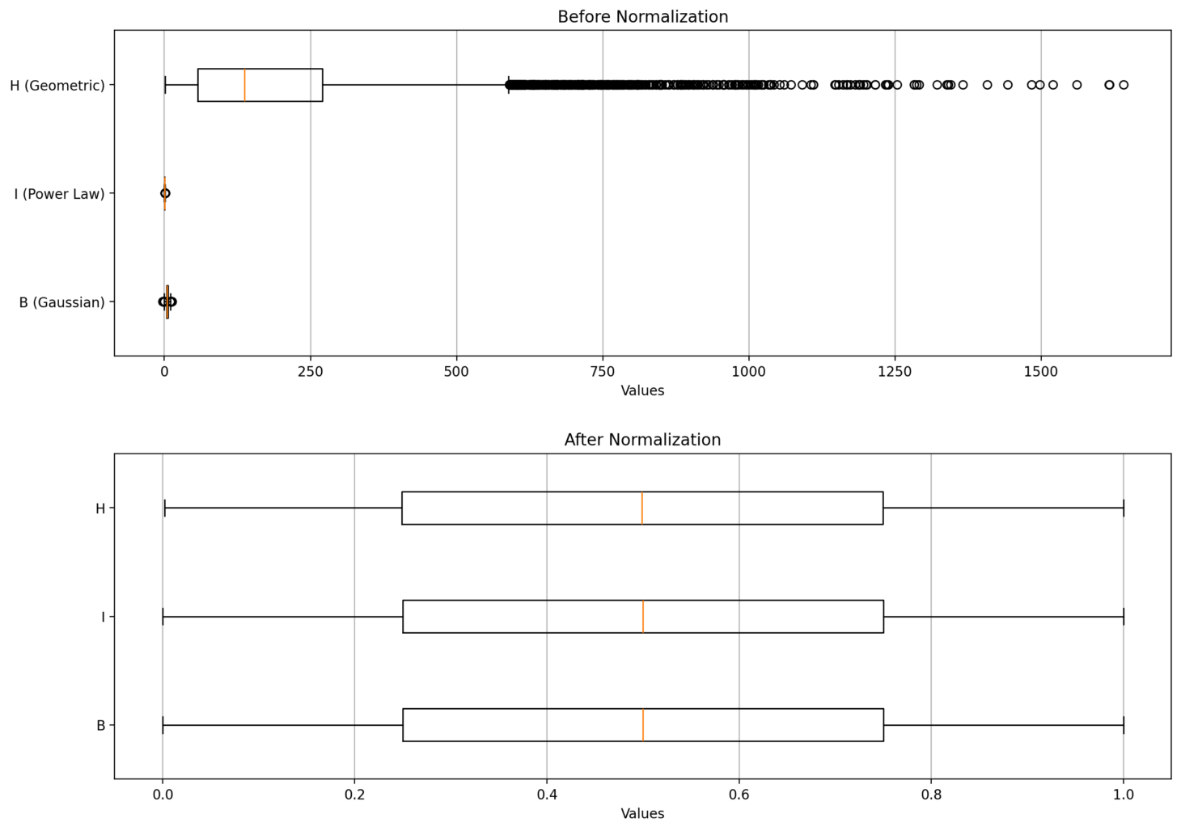


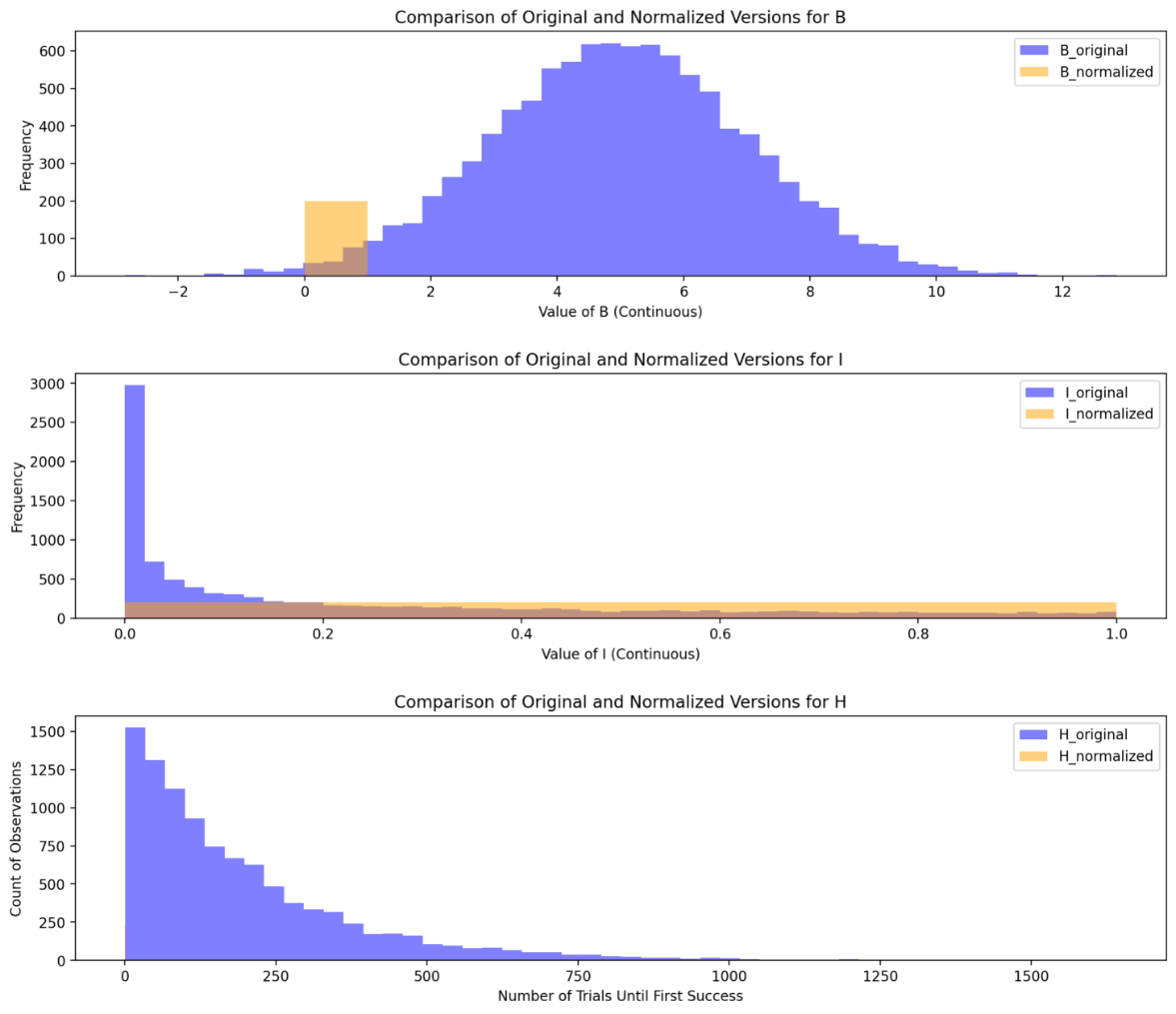
B: Shape remains unchanged but centres it at 0 and scales it to standard deviation of 1.

I: Mean is close to large values thus normalizing them values fall below 0 but skewness is not removed.

H: Values are scaled down significantly but shape still remains-skewed.

**Percentile Scaling:**



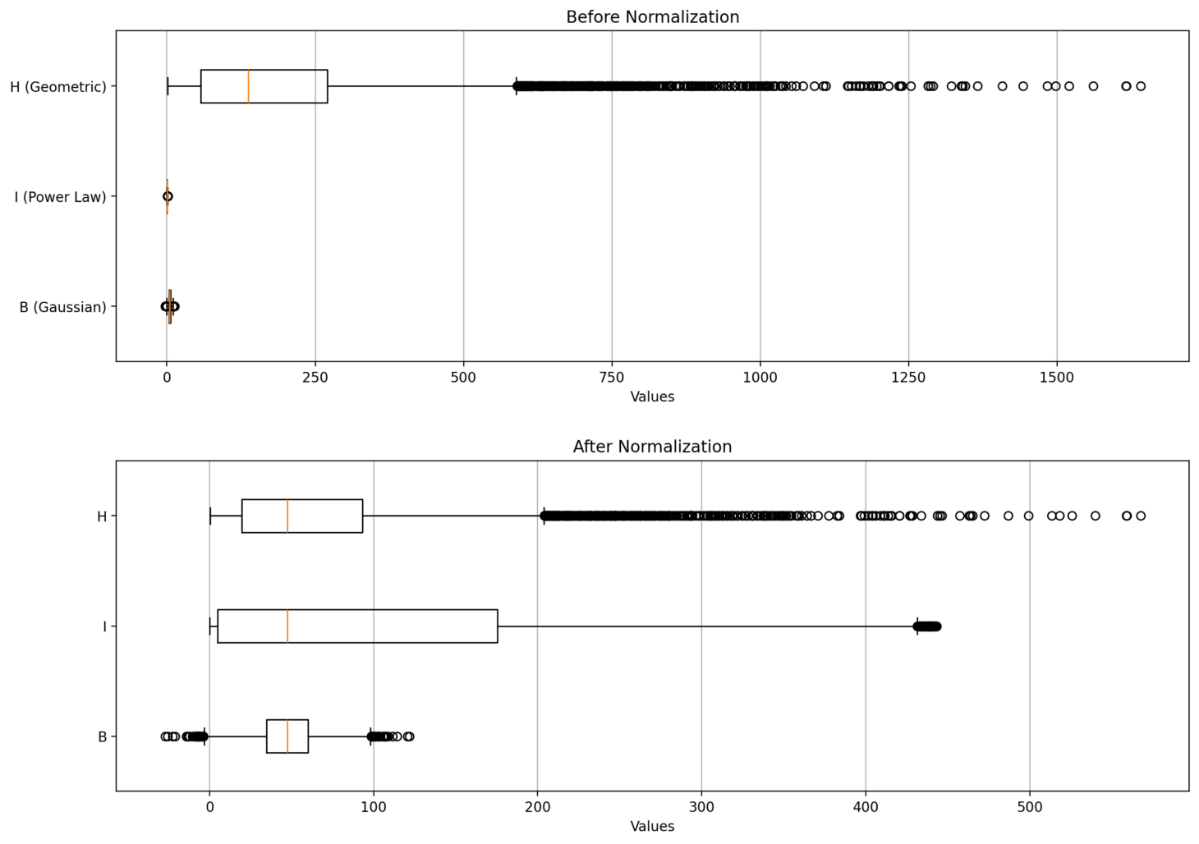


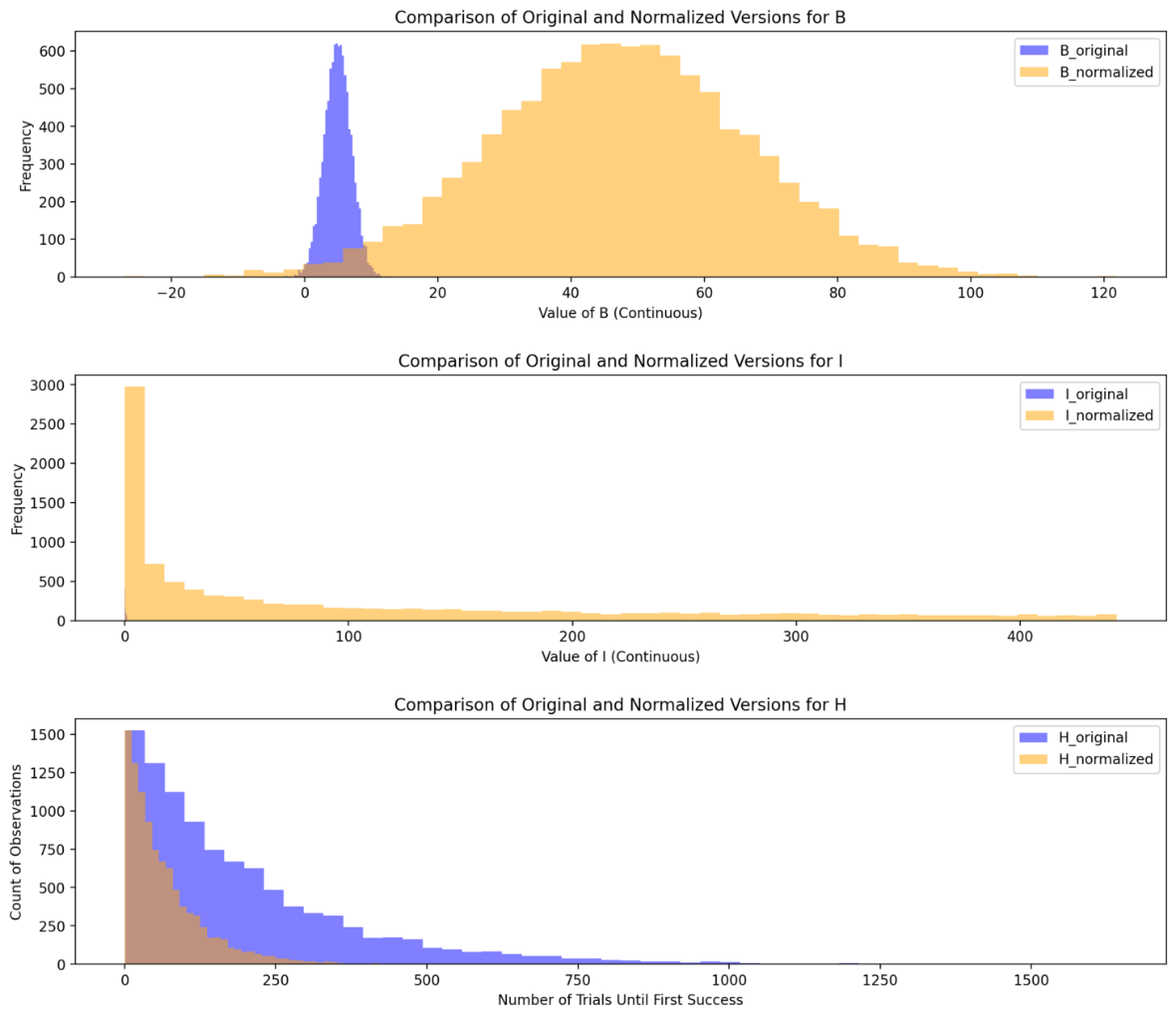
B: Distribution gets transformed into uniform.

I: There are many small values and a few extreme large values. Small values are pushed apart and long tail is compressed making distribution uniform.

H: Skewness is reduced and data is made more continuous.

**Median Normalization:**





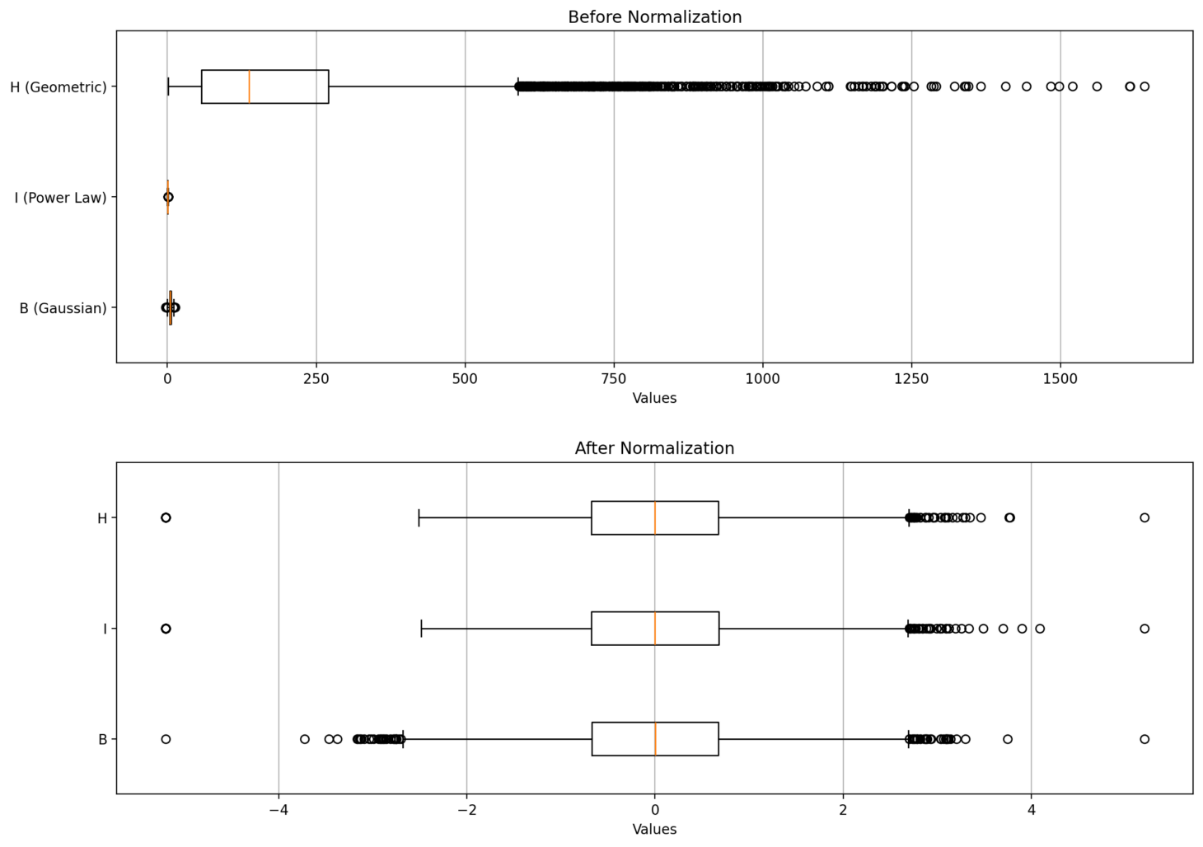
B: Doesn’t change shape but scales values.

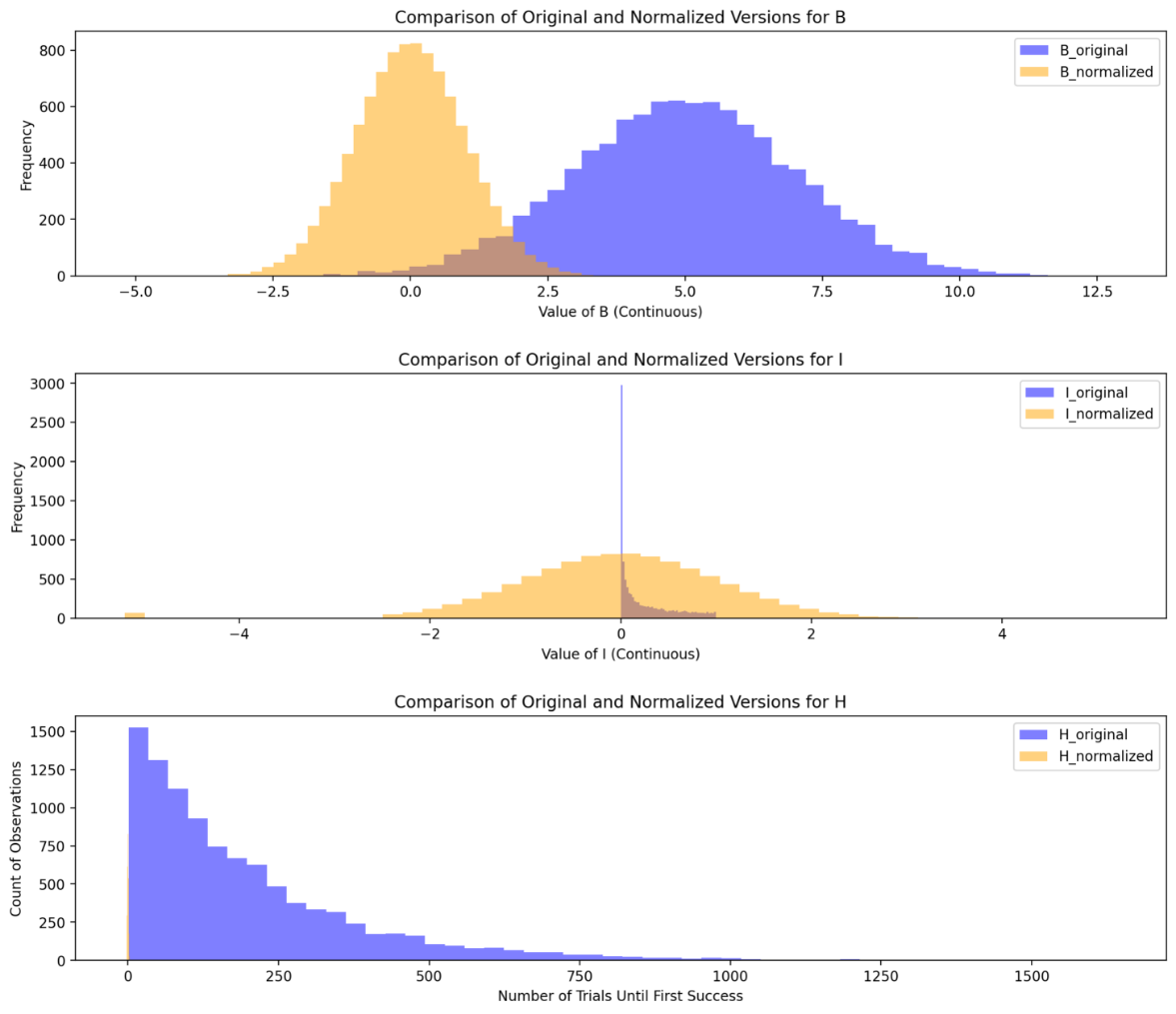
I: Scales the values to match the median but distribution remains the same.

H: Values are scales down and slightly reduces the skewness.

Useful when ranks are important but relative distribution should not be changed.

**Quantile Normalization:**



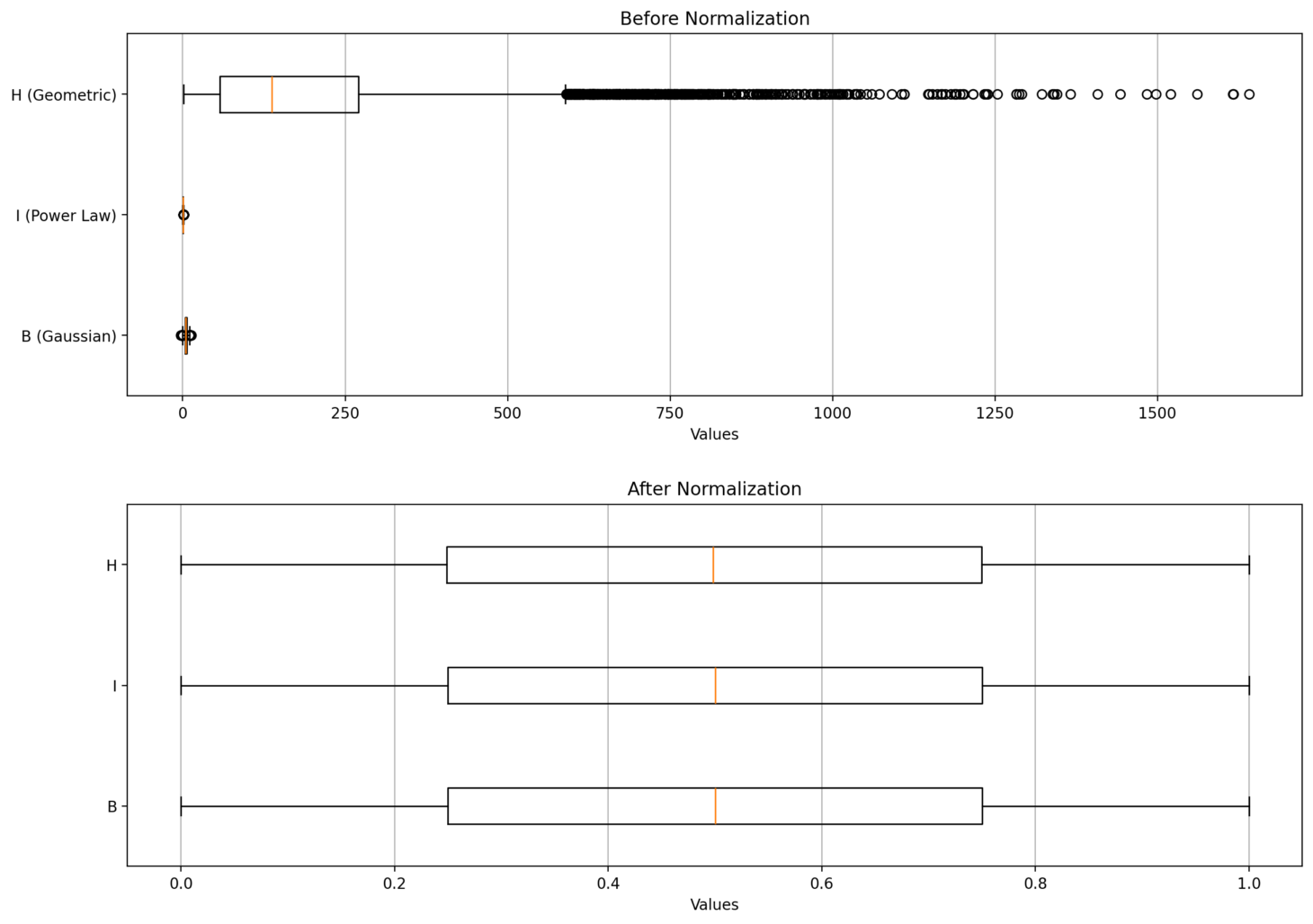


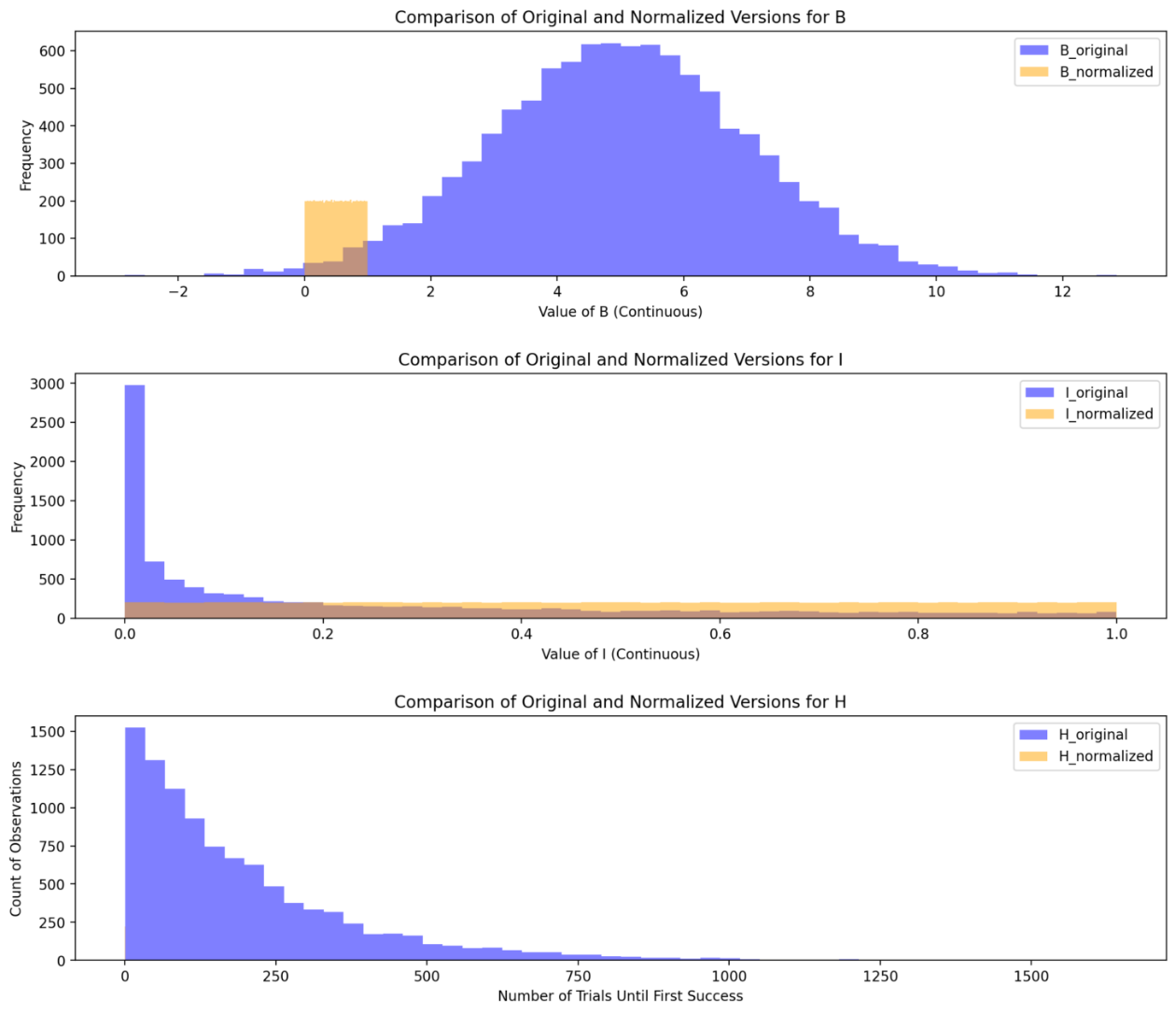
For normal output distribution:

B: Shape is preserved but extreme vales are mapped closer to mean.

I: Transforms into a symmetric normal shape.

H: Data converts into continuous normal distribution.





For uniform output distribution:

B: Shape disappears spreading values evenly between 0 and 1.

I: Large values are compressed and small values are stretched out evenly forcing uniform distribution.

H: Data becomes continuous and spread out evenly.