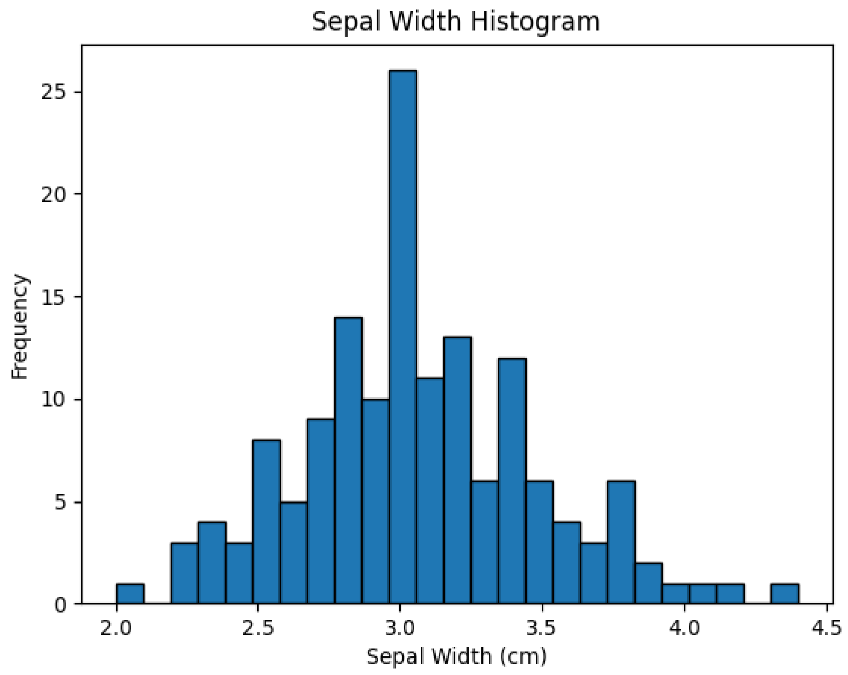
Week 3 Assignment 1

Andrea Del Guidice
1/30/2025

1. Using the `iris` dataset...

   a. Make a histogram of the variable `Sepal.Width`.

```
sepal_widths = irisdf['sepal width (cm)']
plt.hist(sepal_widths, edgecolor='black', bins=25)
plt.xlabel('Sepal Width (cm)')
plt.ylabel('Frequency')
plt.title('Sepal Width Histogram')
plt.show()
```



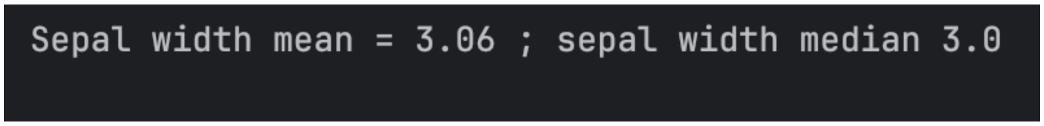
   b. Based on the histogram from #1a, which would you expect to be higher, the mean or the median? Why?

   I struggled to determine skewness from the histogram and explored the data using several different bin sizes (pictured is bins=25). The data is somewhat symmetric but seems to have a slight right tail (i.e., right skew); therefore, I'd expect the mean to be slightly higher than the median.

   c. Confirm your answer to #1b by actually finding these values.

   - Mean is > Median (slight right skew), as 3.06 slightly greater than 3.0:

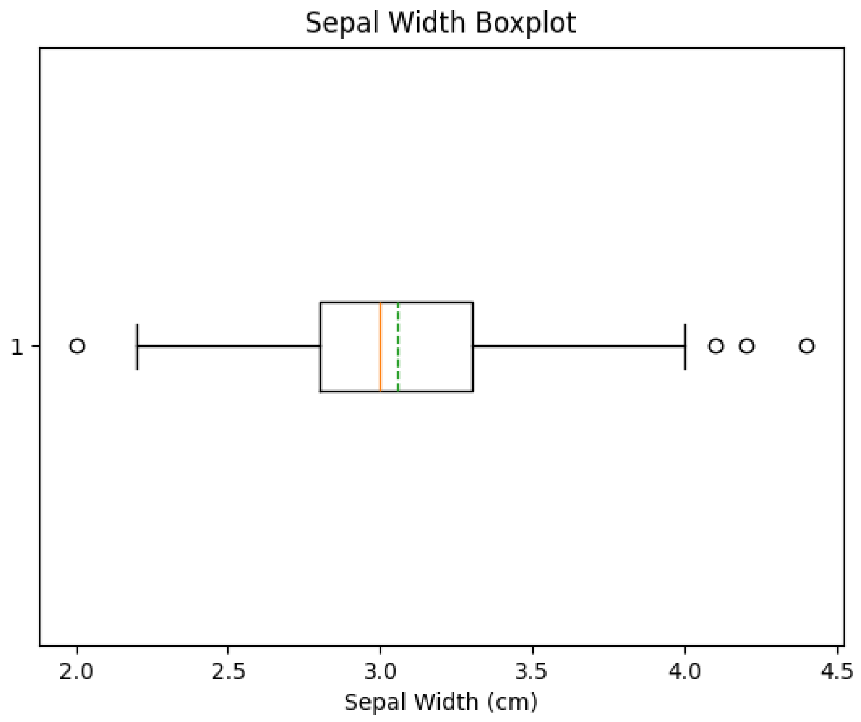
```
sw_mean = sepal_widths.mean()
sw_median = sepal_widths.median()
print("Sepal width mean =",f"{sw_mean:.2f}",";  sepal width
median", sw_median)
```



```
Sepal width mean = 3.06 ; sepal width median 3.0
```

- It also helped me to confirm the right skew using a boxplot (mean > median, with median=red line & mean=green dashed line):

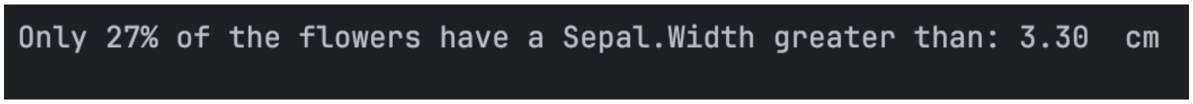
```
#More exploration of Q1 skew
plt.boxplot(sepal_widths, vert=False, showmeans=True,
meanline=True)
plt.title('Sepal Width Boxplot')
plt.xlabel('Sepal Width (cm)')
plt.show()
```



d. Only 27% of the flowers have a Sepal.Width higher than __3.3__ cm.

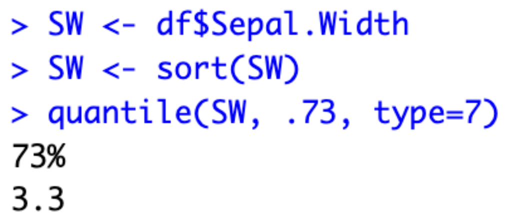
```
# Calculate the sepal width 'x' that 27% of the sepals are greater
# than.  If 27% of flowers have sepals wider than x, 100-27 = 73%
# are *as or more* narrow (i.e., the 73rd percentile):
```

```
width_boundary = np.percentile(sepal_widths, 73)
print("Only 27% of the flowers have a Sepal.Width greater
than:",f"{width_boundary:.2f}"," cm")
```



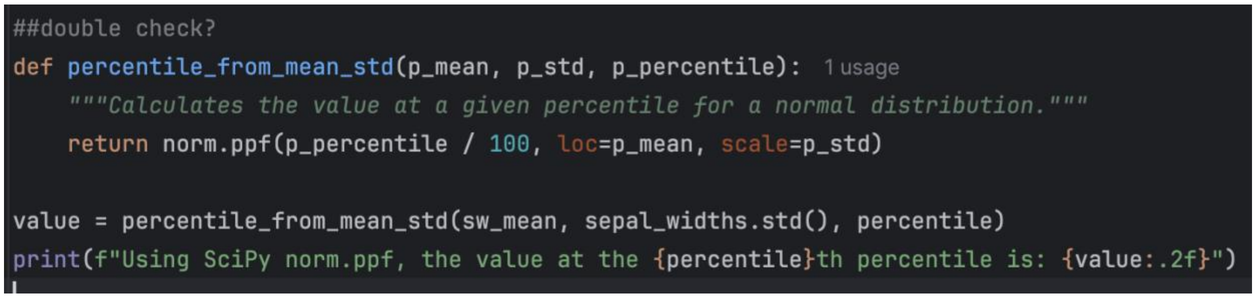
```
Only 27% of the flowers have a Sepal.Width greater than: 3.30  cm
```

Note: I went a little down a rabbit whole investigating how to verify my answer. I learned there are multiple formulas for calculating percentiles/quantiles, although all methods tried seemed to return same 3.3 answer in both Python and R (different data sets did return very different answers per method, though).

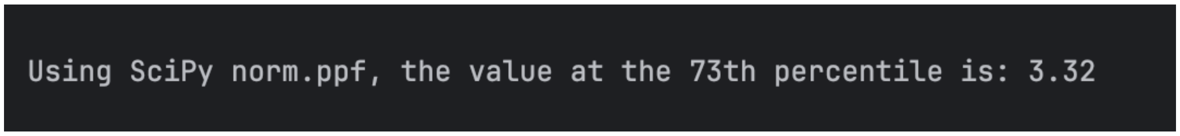


```
> SW <- df$Sepal.Width
> SW <- sort(SW)
> quantile(SW, .73, type=7)
73%
3.3
```

I'm still wondering how to best validate my answer to this Q1.d though? I tried using the SciPy function below to validate and it was close, but I also thought it was unreasonable I should expect this function to return the same answer as above...



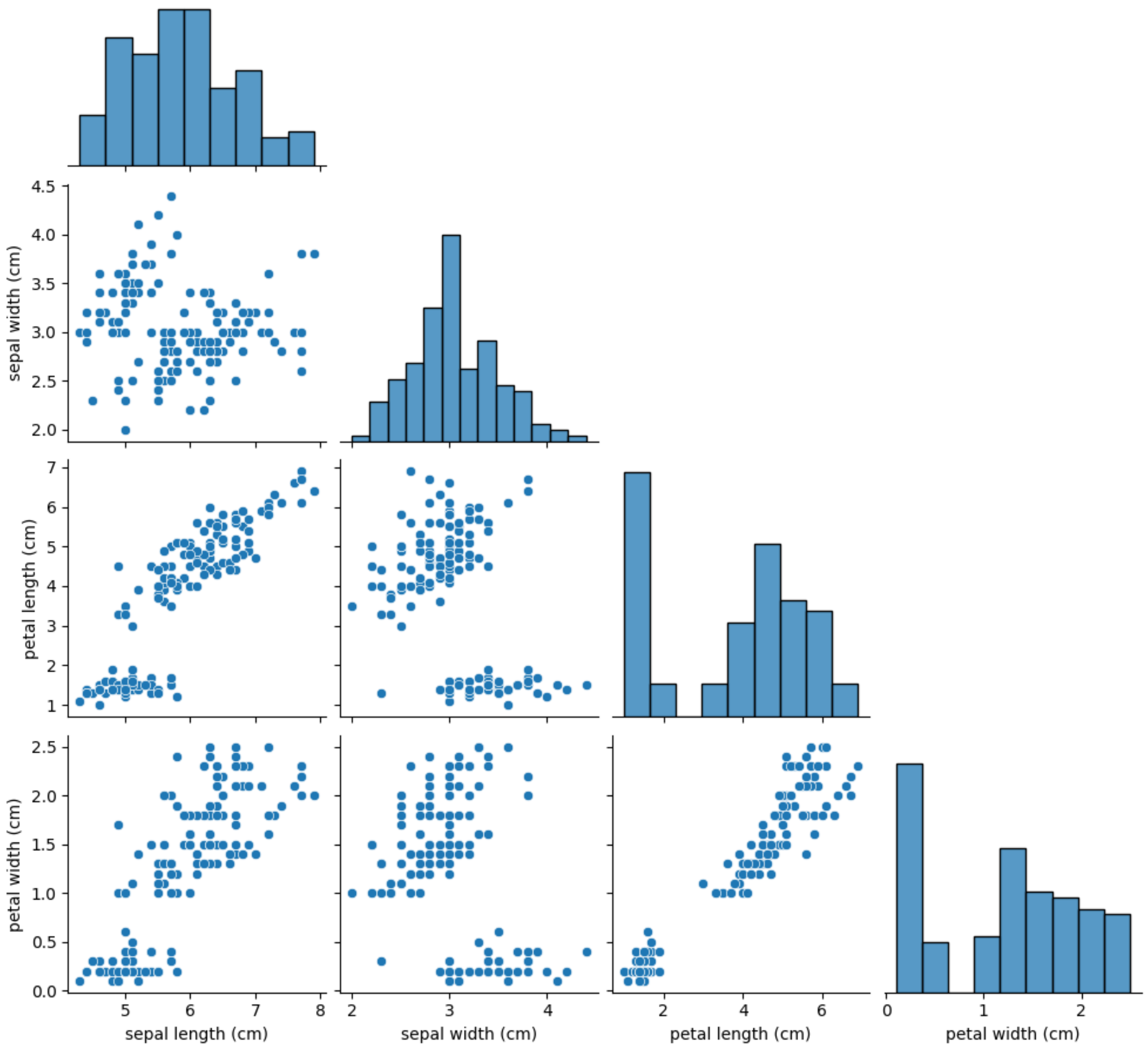
```
##double check?
def percentile_from_mean_std(p_mean, p_std, p_percentile):  1 usage
    """Calculates the value at a given percentile for a normal distribution."""
    return norm.ppf(p_percentile / 100, loc=p_mean, scale=p_std)

value = percentile_from_mean_std(sw_mean, sepal_widths.std(), percentile)
print(f"Using SciPy norm.ppf, the value at the {percentile}th percentile is: {value:.2f}")
```



```
Using SciPy norm.ppf, the value at the 73th percentile is: 3.32
```

e. Make scatterplots of each pair of the numerical variables in iris (There should be 6 pairs/plots).

```
# Create a pairplot using seaborn
sns.pairplot(irisdf, vars=irisdf.columns[:4], kind="scatter",
corner=True)
plt.show()
```



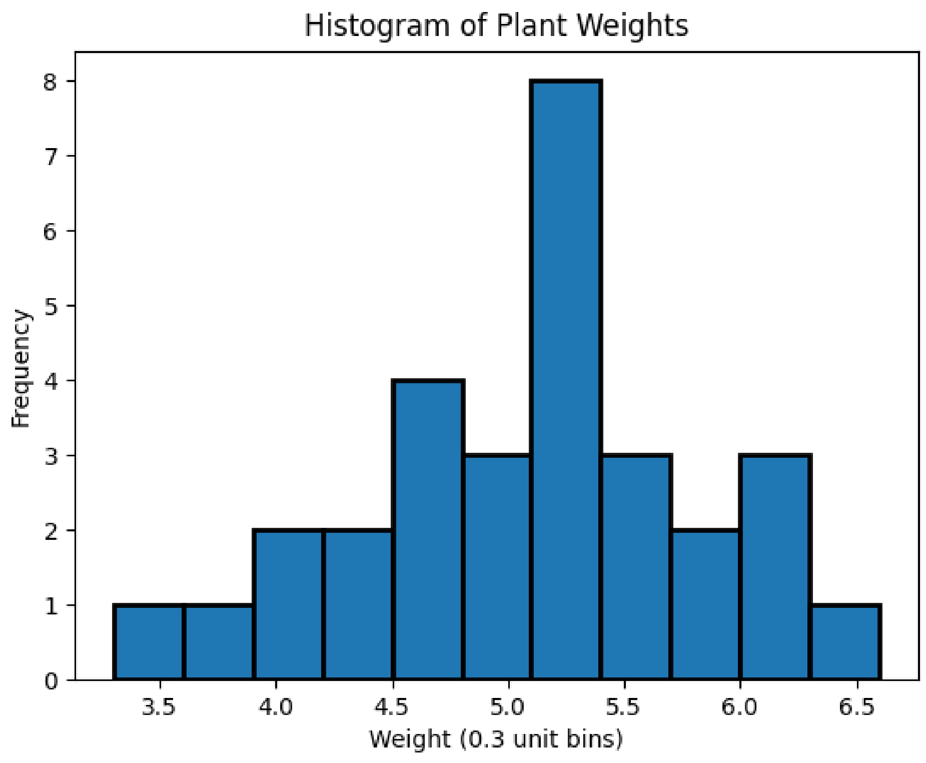
f. Based on #1e, which two variables appear to have the strongest relationship? And which two appear to have the weakest relationship?

Petal length and petal width appear to have the strongest relationship. Sepal length and sepal width seem to have the weakest relationship.

2. Using the `PlantGrowth` dataset...

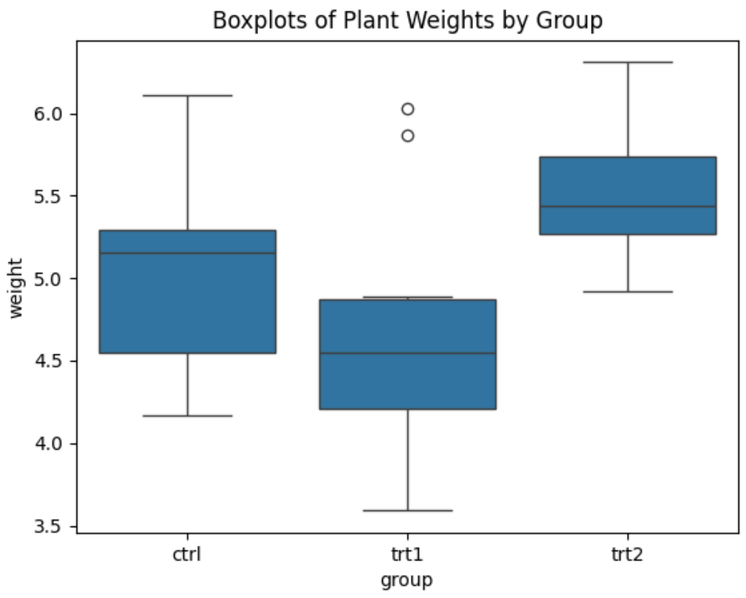
a. Make a histogram of the variable `weight` with breakpoints (bin edges) at every 0.3 units, starting at 3.3.

```
bin_start = 3.3
bin_step = 0.3
weights = PlantGrowth['weight']
#make sure we include all values in histogram
max_edge = weights.max() + bin_step
xbins = np.arange(bin_start, max_edge, bin_step)
style = {'edgecolor': 'black', 'linewidth': 2}

fig, ax = plt.subplots()
ax.hist(weights, bins=xbins, **style)
ax.set_ylabel('Frequency')
ax.set_xlabel('Weight (0.3 unit bins)')
ax.set_title('Histogram of Plant Weights')
plt.show()
```



b. Make boxplots of `weight` separated by `group` in a single graph.

```
sns.boxplot(data=PlantGrowth, x="group", y="weight")
plt.title('Boxplots of Plant Weights by Group')
plt.show()
```

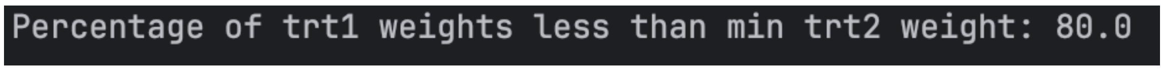


c. Based on the boxplots in #2b, approximately what percentage of the "trt1" `weight`s are below the minimum "trt2" `weight`?

I see two outliers in group trt1 laying above the lower whisker (min) of trt2 and I know there are 10 plants within each group, so my guesstimate is 8/10 or 80%. I wouldn't have been able to guess that, however, without knowing how many datapoints were in each group—I can't tell count/frequency from this graph.

d. Find the exact percentage of the "trt1" `weight`s that are below the minimum "trt2" `weight`. Answer = 80%

```
trt2_weights = PlantGrowth['weight'][PlantGrowth['group']=='trt2']
trt2_min = np.min(trt2_weights)
trt1_weights = PlantGrowth['weight'][PlantGrowth['group']=='trt1']
trt1_count = trt1_weights.count()
#get count of just those trt1 weights less than min trt2 weight
trt1_belowtrt2_count = trt1_weights[trt1_weights < trt2_min].count()
percentage_trt1_below_trt2 = (trt1_belowtrt2_count/trt1_count)*100
print("Percentage of trt1 weights less than min trt2 weight:",
percentage_trt1_below_trt2)
```

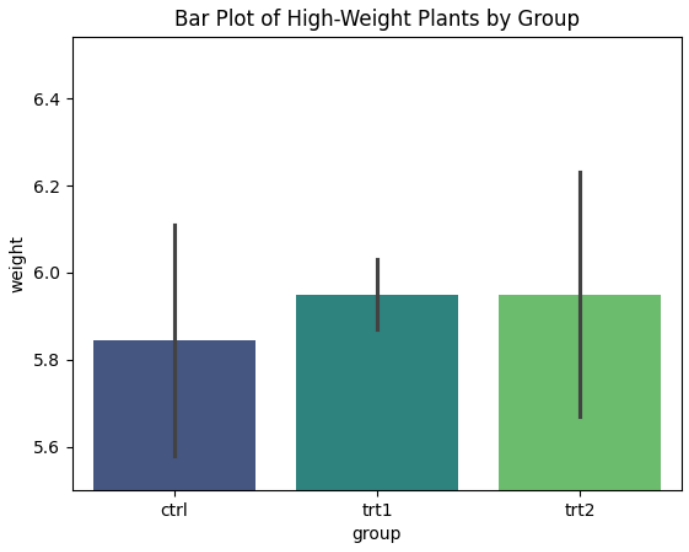


```
Percentage of trt1 weights less than min trt2 weight: 80.0
```

e. Only including plants with a weight above 5.5, make a barplot of the variable `group`. Make the barplot colorful using some color palette.

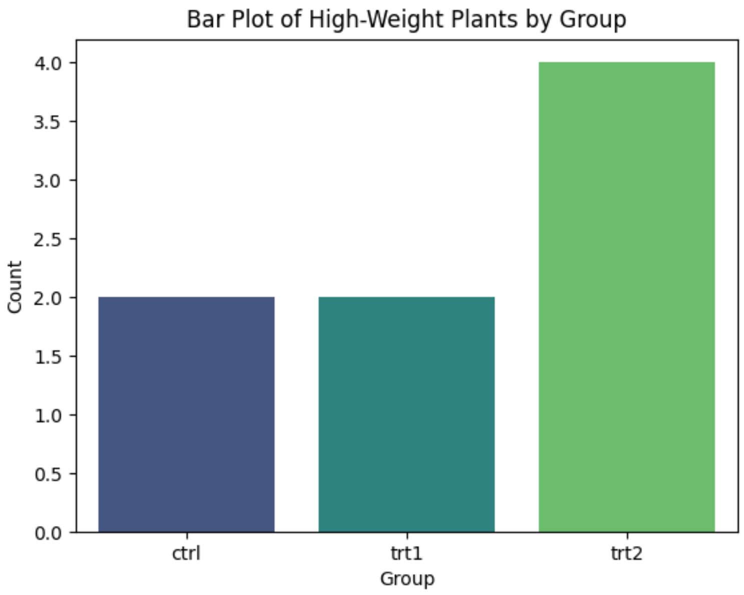
- First I created a bar plot of plants weighing more than 5.5 by group (x=group, y=weight):

```
weight_limit = 5.5
hw_plants = PlantGrowth[PlantGrowth['weight'] > weight_limit]
hw_groups = hw_plants['group']
hw_weights = hw_plants['weight']
sns.barplot(x=hw_groups, y=hw_weights, palette="viridis")
plt.ylim(weight_limit)#start plot at 5.5
plt.title('Bar Plot of High-Weight Plants by Group')
plt.show()
```



- I found this plot at best unclear, however, so I next created a bar plot that displayed how many high-weight (i.e., weight above 5.5) plants there were in each group (below). This plot told a clearer story from the data, that treatment 2 group had more high-weight plants that the other two groups:

```
##Plot 2 - told a clearer story
weight_limit = 5.5
hw_plants = PlantGrowth[PlantGrowth['weight'] >
weight_limit].sort_values(by='group')
frequency_table = hw_plants['group'].value_counts().sort_values()
labels = frequency_table.index
values = frequency_table.values

# Create the bar plot
sns.barplot(x=labels, y=values, palette="viridis")
plt.title("Barplot of High Weight Plants by Group")
plt.xlabel("Group")
plt.ylabel("Count")
plt.show()
```


Bar Plot of High-Weight Plants by Group
Count
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
ctrl
trt1
trt2
Group