Week_3_Assignment

September 14, 2025

1 Week 3 Assignment

[1]:
```
from sklearn import datasets
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
data = { "weight": [4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14,␣
 ↪4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69, 6.31, 5.12, 5.
 ↪54, 5.50, 5.37, 5.29, 4.92, 6.15, 5.80, 5.26], "group": ["ctrl"] * 10 +␣
 ↪["trt1"] * 10 + ["trt2"] * 10}
PlantGrowth = pd.DataFrame(data)
```

[3]:
```
iris = datasets.load_iris()

iris_df = pd.DataFrame(iris.data, columns=iris.feature_names)
iris_df['target']=iris.target
iris_df['target_name'] = np.where(iris_df['target']==0,'setosa',np.
 ↪where(iris_df['target']==1,'versicolor','virginica'))
```

[4]:
```
iris_df
```

[4]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	\
0	5.1	3.5	1.4	0.2	
1	4.9	3.0	1.4	0.2	
2	4.7	3.2	1.3	0.2	
3	4.6	3.1	1.5	0.2	
4	5.0	3.6	1.4	0.2	
..	…	…	…	…	
145	6.7	3.0	5.2	2.3	
146	6.3	2.5	5.0	1.9	
147	6.5	3.0	5.2	2.0	
148	6.2	3.4	5.4	2.3	
149	5.9	3.0	5.1	1.8	

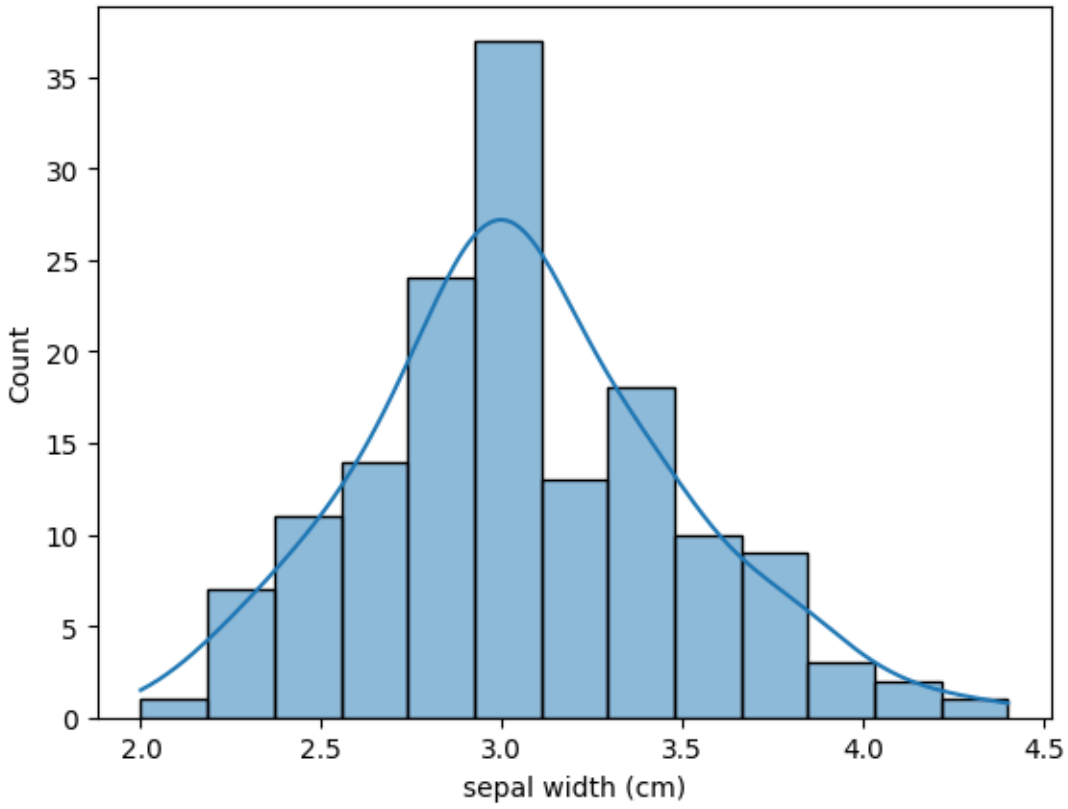
	target	target_name

```
0          0      setosa
1          0      setosa
2          0      setosa
3          0      setosa
4          0      setosa
..       …          …
145        2   virginica
146        2   virginica
147        2   virginica
148        2   virginica
149        2   virginica

[150 rows x 6 columns]
```

1.1 1. Using the iris dataset...

1.1.1 1.1 Make a histogram of the variable Sepal.Width.

[6]:
```
sns.histplot(x="sepal width (cm)", data=iris_df, kde=True)

plt.show()
```



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

Based on the distribution, it looks pretty close to the normal distribution, so it seems that median and mean might be very close. Although the distribution is a little right-skewed, so probably the mean is little higher.

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

[9]:
```
print(f'''The sepal width has the following distribution mean {iris_df['sepal␣
↪width (cm)'].mean()} and the median is: {iris_df['sepal width (cm)'].
↪median()}
      ''')
```

```
The sepal width has the following distribution mean 3.0573333333333337 and the
median is: 3.0
```

1.1.4 Only 27% of the flowers have a Sepal.Width higher than _____ cm.

[10]:
```
iris_df['sepal width (cm)'].quantile(0.73)
```

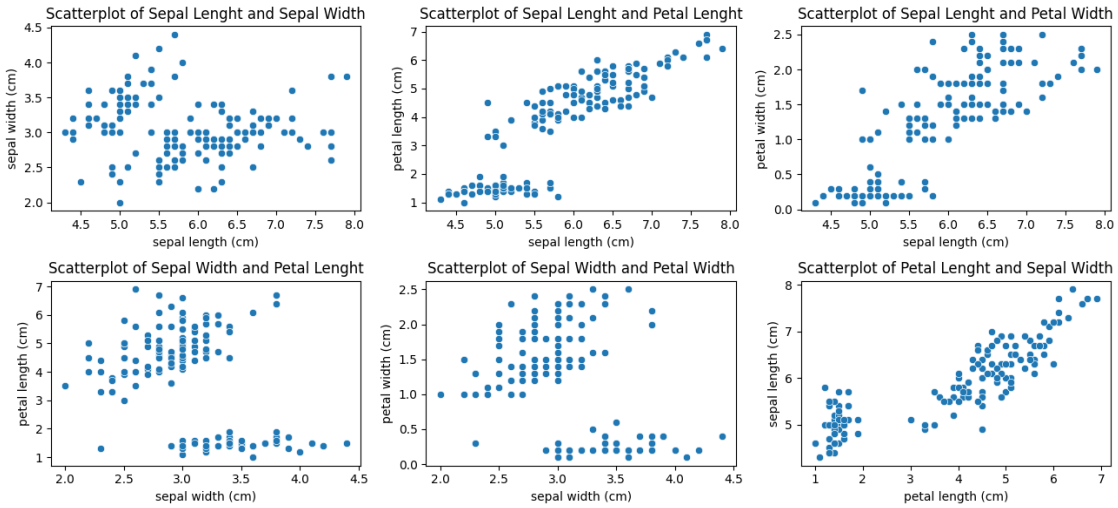
[10]: 3.3

The question we are looking to answer is which is the cm lenght, from which only 27% of the flowers have a higher width. To answer this, we can use the quantile, in this case 0.73 quantile (as quantile measure the values under the 73% of the population if lower).

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

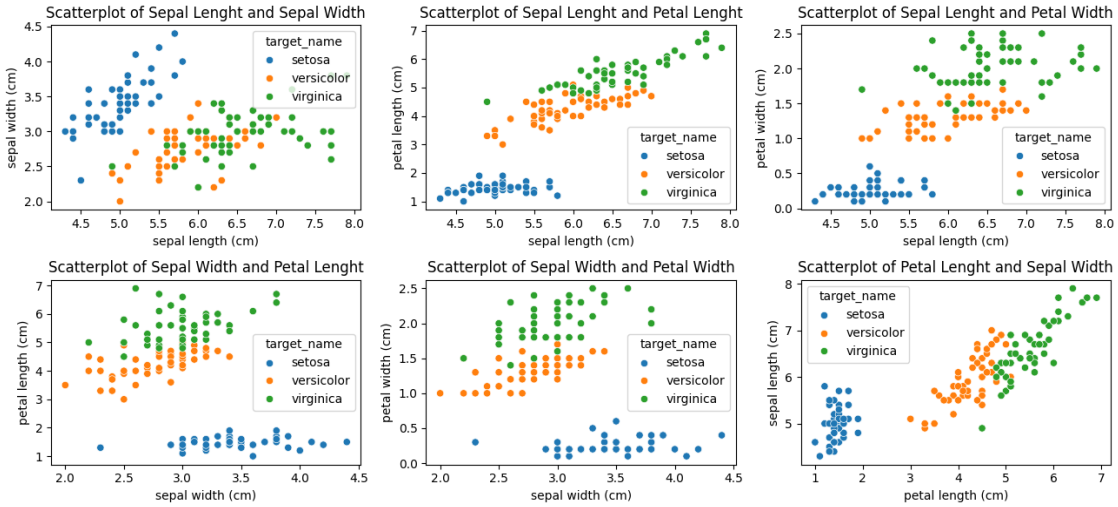
[24]:
```
# Create a figure with a 1x2 grid of subplots
fig, axes = plt.subplots(2, 3, figsize=(13, 6))

sns.scatterplot(x="sepal length (cm)", y="sepal width (cm)", data=iris_df,␣
↪ax=axes[0,0])
axes[0,0].set_title('Scatterplot of Sepal Lenght and Sepal Width')

sns.scatterplot(x="sepal length (cm)", y="petal length (cm)", data=iris_df,␣
↪ax=axes[0,1])
axes[0,1].set_title('Scatterplot of Sepal Lenght and Petal Lenght')

sns.scatterplot(x="sepal length (cm)", y="petal width (cm)", data=iris_df,␣
↪ax=axes[0,2])
axes[0,2].set_title('Scatterplot of Sepal Lenght and Petal Width')

sns.scatterplot(x="sepal width (cm)", y="petal length (cm)", data=iris_df,␣
↪ax=axes[1,0])
```

```
axes[1,0].set_title('Scatterplot of Sepal Width and Petal Lenght')

sns.scatterplot(x="sepal width (cm)", y="petal width (cm)", data=iris_df,␣
 ↪ax=axes[1,1])
axes[1,1].set_title('Scatterplot of Sepal Width and Petal Width')

sns.scatterplot(x="petal length (cm)", y="sepal length (cm)", data=iris_df,␣
 ↪ax=axes[1,2])
axes[1,2].set_title('Scatterplot of Petal Lenght and Sepal Width')


plt.tight_layout() # Adjust layout to prevent overlapping
plt.show()
```



[22]:
```
# Create a figure with a 1x2 grid of subplots
fig, axes = plt.subplots(2, 3, figsize=(13, 6))

sns.scatterplot(x="sepal length (cm)", y="sepal width (cm)", data=iris_df,␣
 ↪ax=axes[0,0], hue="target_name")
axes[0,0].set_title('Scatterplot of Sepal Lenght and Sepal Width')

sns.scatterplot(x="sepal length (cm)", y="petal length (cm)", data=iris_df,␣
 ↪ax=axes[0,1], hue="target_name")
axes[0,1].set_title('Scatterplot of Sepal Lenght and Petal Lenght')

sns.scatterplot(x="sepal length (cm)", y="petal width (cm)", data=iris_df,␣
 ↪ax=axes[0,2], hue="target_name")
axes[0,2].set_title('Scatterplot of Sepal Lenght and Petal Width')
```

```
sns.scatterplot(x="sepal width (cm)", y="petal length (cm)", data=iris_df,␣
 ↪ax=axes[1,0], hue="target_name")
axes[1,0].set_title('Scatterplot of Sepal Width and Petal Lenght')

sns.scatterplot(x="sepal width (cm)", y="petal width (cm)", data=iris_df,␣
 ↪ax=axes[1,1], hue="target_name")
axes[1,1].set_title('Scatterplot of Sepal Width and Petal Width')

sns.scatterplot(x="petal length (cm)", y="sepal length (cm)", data=iris_df,␣
 ↪ax=axes[1,2], hue="target_name")
axes[1,2].set_title('Scatterplot of Petal Lenght and Sepal Width')


plt.tight_layout() # Adjust layout to prevent overlapping
plt.show()
```



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

[18]:
```
iris_df[['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)','petal␣
 ↪width (cm)']].corr()
```

[18]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
sepal length (cm)	1.000000	-0.117570	0.871754	0.817941
sepal width (cm)	-0.117570	1.000000	-0.428440	-0.366126
petal length (cm)	0.871754	-0.428440	1.000000	0.962865
petal width (cm)	0.817941	-0.366126	0.962865	1.000000

```
sepal length (cm)          0.817941
sepal width (cm)          -0.366126
petal length (cm)          0.962865
petal width (cm)           1.000000
```

[19]:
```
iris_df[iris_df['target_name']=='setosa'][['sepal length (cm)', 'sepal width
↪(cm)', 'petal length (cm)','petal width (cm)']].corr()
```

[19]:
```
                   sepal length (cm)  sepal width (cm)  petal length (cm)  \
sepal length (cm)           1.000000          0.742547           0.267176
sepal width (cm)            0.742547          1.000000           0.177700
petal length (cm)           0.267176          0.177700           1.000000
petal width (cm)            0.278098          0.232752           0.331630

                   petal width (cm)
sepal length (cm)          0.278098
sepal width (cm)           0.232752
petal length (cm)          0.331630
petal width (cm)           1.000000
```

[20]:
```
iris_df[iris_df['target_name']=='versicolor'][['sepal length (cm)', 'sepal
↪width (cm)', 'petal length (cm)','petal width (cm)']].corr()
```

[20]:
```
                   sepal length (cm)  sepal width (cm)  petal length (cm)  \
sepal length (cm)           1.000000          0.525911           0.754049
sepal width (cm)            0.525911          1.000000           0.560522
petal length (cm)           0.754049          0.560522           1.000000
petal width (cm)            0.546461          0.663999           0.786668

                   petal width (cm)
sepal length (cm)          0.546461
sepal width (cm)           0.663999
petal length (cm)          0.786668
petal width (cm)           1.000000
```

[21]:
```
iris_df[iris_df['target_name']=='virginica'][['sepal length (cm)', 'sepal width
↪(cm)', 'petal length (cm)','petal width (cm)']].corr()
```

[21]:
```
                   sepal length (cm)  sepal width (cm)  petal length (cm)  \
sepal length (cm)           1.000000          0.457228           0.864225
sepal width (cm)            0.457228          1.000000           0.401045
petal length (cm)           0.864225          0.401045           1.000000
petal width (cm)            0.281108          0.537728           0.322108

                   petal width (cm)
sepal length (cm)          0.281108
sepal width (cm)           0.537728
```

```
petal length (cm)          0.322108
petal width (cm)           1.000000
```

As shown in the initial plot, the pair petal width (cm) vs. petal length (cm) has the strongest relationship. Although the plot shows clusters, this suggests that other variables may influence the relationship.

Based on that assumption, we can plot the data by flower type, which reveals an interesting story: this strong association only holds for Versicolors.

To better understand the relationship, we can use the correlation matrix of both the whole dataset and the data partitioned by flower species.

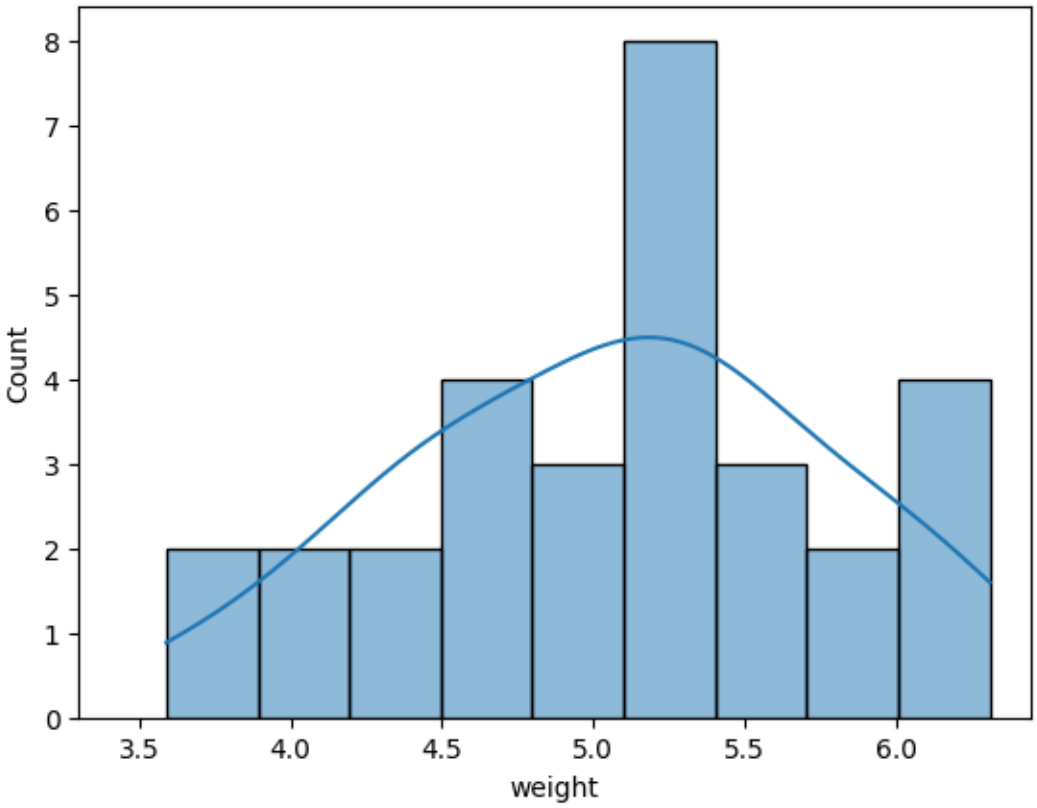
The results are consistent with our conclusions from analyzing the plots. In the whole dataset, the strongest relationship is between petal length and width. In Setosa, however, the strongest relationship is between sepal length and width. In Versicolor, most variables are well correlated, but the most important feature is again petal length and width. Finally, in Virginica, the strongest association is between sepal length and petal length.

This confirms that the relationships between variables differ by flower species, indicating mixed effects. Drawing a conclusion without considering this effect could lead to misleading results.

1.2 2. Using the PlantGrowth dataset...

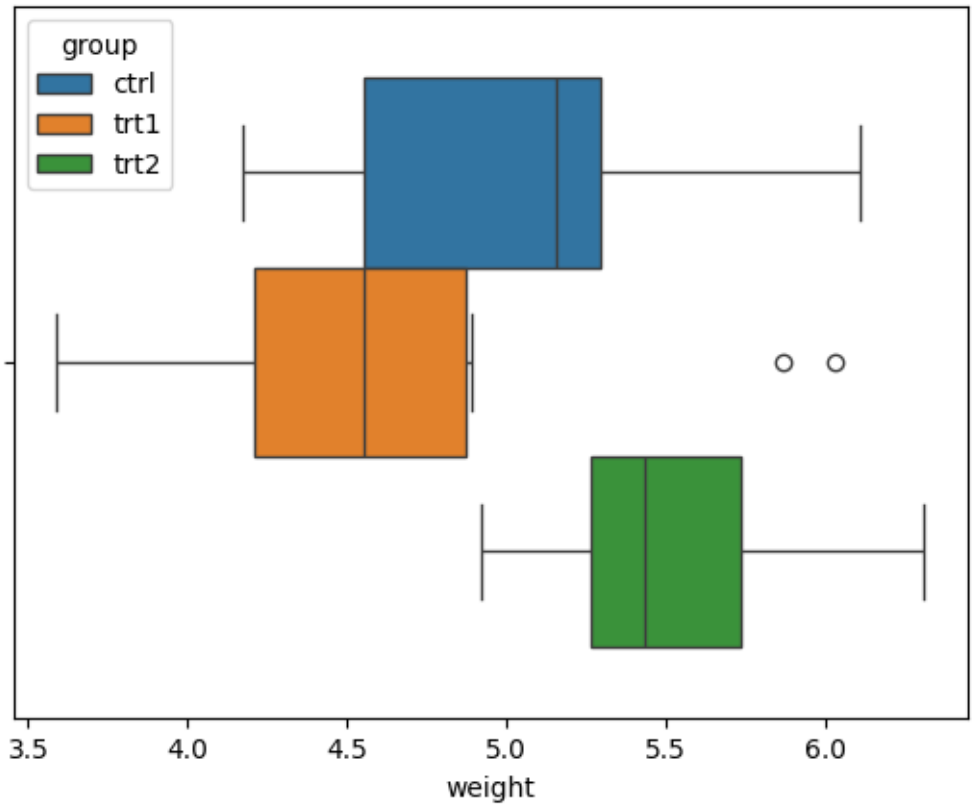
1.2.1 Make a histogram of the variable weight with breakpoints (bin edges) at every 0.3 units, starting at 3.3.

[33]:
```
ax = sns.histplot(x="weight", data=PlantGrowth, binwidth=0.3, kde=True)
ax.set_xlim(left=3.3)
plt.show()
```



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

[35]:
```
sns.boxplot(x="weight", data=PlantGrowth, hue='group')
plt.show()
```



1.2.3 Based on the boxplots in #2b, approximately what percentage of the "trt1" weights are below the minimum "trt2" weight?

Based on the plot of both trt1 and trt2, only two outlier points of trt1 are greater than the left part of the box plot of trt2, I estimate this to be approximatily the 90% percentile

1.2.4 Find the exact percentage of the "trt1" weights that are below the minimum "trt2" weight.

[44]:
```
PlantGrowth[PlantGrowth['group']=='trt1'].size
```

[44]:
```
20
```

[38]:
```
PlantGrowth[PlantGrowth['group']=='trt1'].describe()
```

[38]:

	weight
count	10.000000
mean	4.661000
std	0.793676
min	3.590000
25%	4.207500

```
50%     4.550000
75%     4.870000
max     6.030000
```

[39]:
```
PlantGrowth[PlantGrowth['group']=='trt2'].describe()
```

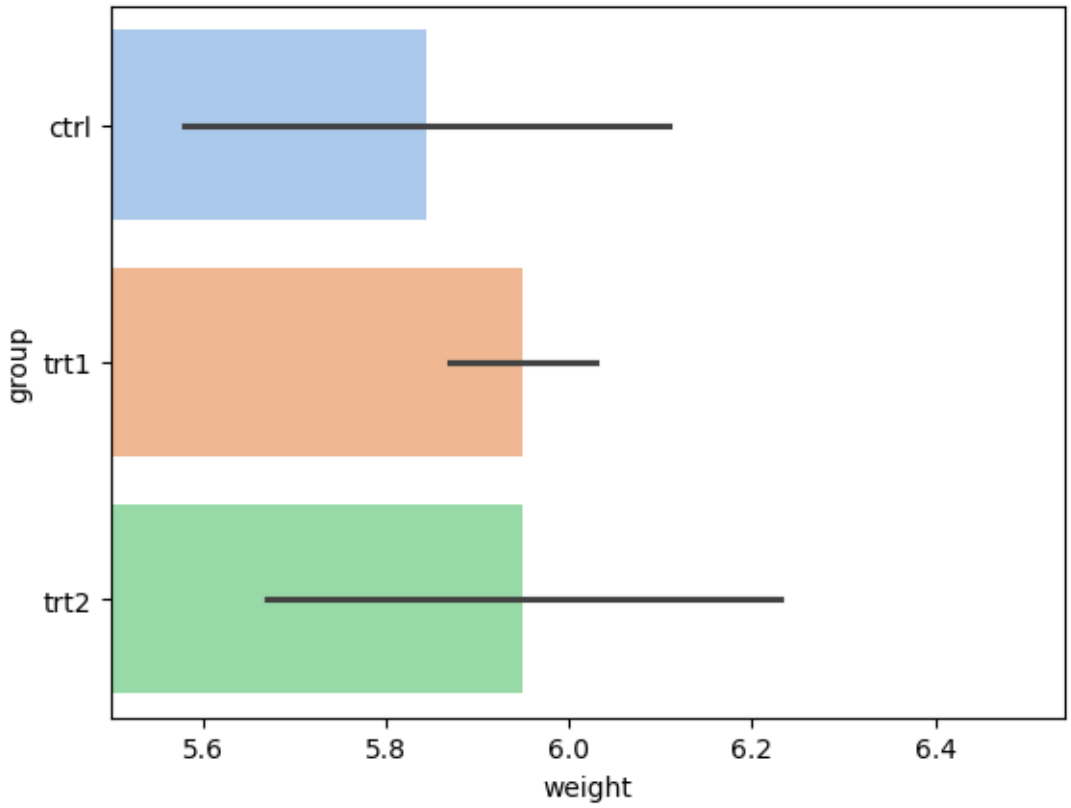
[39]:
```
          weight
count  10.000000
mean    5.526000
std     0.442573
min     4.920000
25%     5.267500
50%     5.435000
75%     5.735000
max     6.310000
```

[42]:
```
from scipy import stats

stats.percentileofscore(PlantGrowth[PlantGrowth['group']=='trt1']['weight'], 4.
↪920000, kind='rank')
```

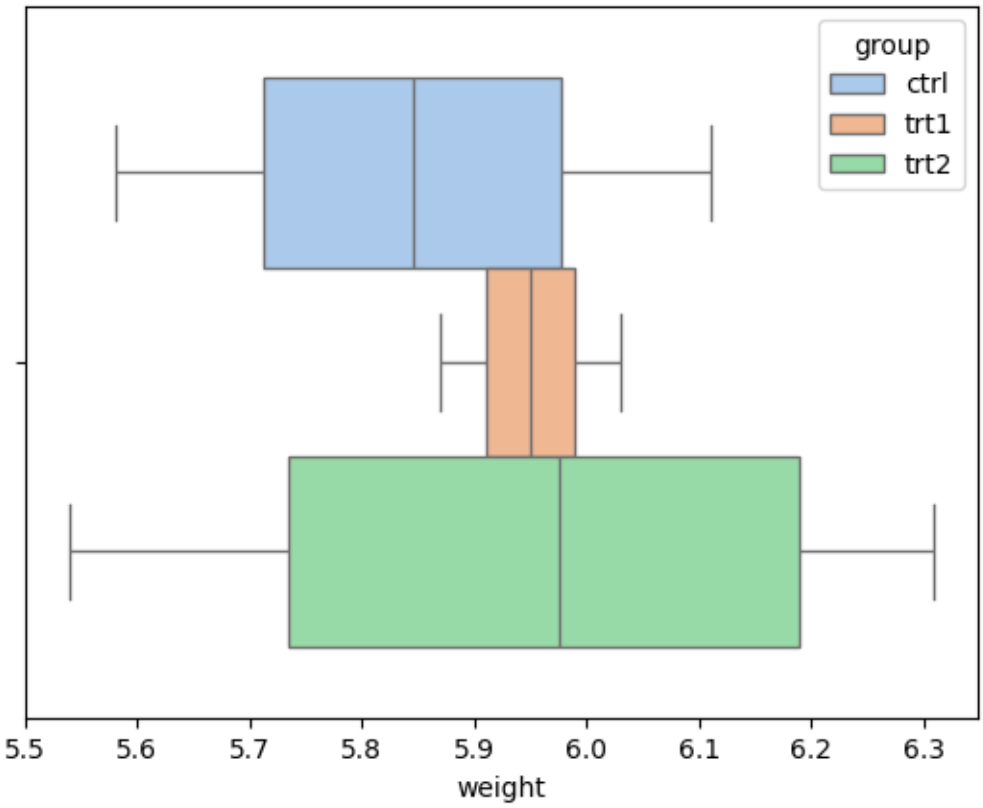
[42]:
```
80.0
```

1.2.5 Only including plants with a weight above 5.5, make a barplot of the variable group. Make the barplot colorful using some color palette (in R, try running ?heat.colors and/or check out https://www.r-bloggers.com/palettes-in-r/).

[51]:
```
sns.set_palette('pastel')
ax=sns.barplot(x="weight", y='group',data=PlantGrowth[PlantGrowth['weight']>5.
↪5],hue='group')
ax.set_xlim(left=5.5)
plt.show()
```



[52]:
```
sns.set_palette('pastel')
ax=sns.boxplot(x="weight",data=PlantGrowth[PlantGrowth['weight']>5.
↪5],hue='group')
ax.set_xlim(left=5.5)
plt.show()
```


group
ctrl
trt1
trt2
5.5
5.6
5.7
5.8
5.9
6.0
6.1
6.2
6.3
weight