Week 3 - Assignment #1

from sklearn import datasets

iris = datasets.load\_iris()

import pandas as pd

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)

print(iris)        # View/Check datasets

print(iris.keys()) # All Dictionary Keys

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 1. Using the iris dataset...

# 1.a - Make a histogram of the variable Sepal.Width.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

# Convert iris to DataFrame

iris\_df = pd.DataFrame(iris.data, columns=iris.feature\_names)

iris\_df["species"] = iris.target

plt.figure(figsize=(7,4))

plt.hist(iris\_df["sepal width (cm)"], bins=10, edgecolor="skyblue")

plt.title("Histogram of Sepal.Width")

plt.xlabel("Sepal.Width (cm)")

plt.ylabel("Count")

plt.show()

A screenshot of a computer

AI-generated content may be incorrect.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

If the histogram is left-skewed(tail to the left): Mean < Median.

If right-skewed(tail to the right): Mean > Median.

If symmetric: Mean=Median.

Observation:

Looking at histogram, the Sepal.Width has longer tail on the right (right-skewed), the mean will be greater than Median.

"""

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mean\_val = iris\_df["sepal width (cm)"].mean()

median\_val = iris\_df["sepal width (cm)"].median()

print(f"Mean:{mean\_val:.2f},Median:{median\_val:.2f}")

A screen shot of a computer

AI-generated content may be incorrect.

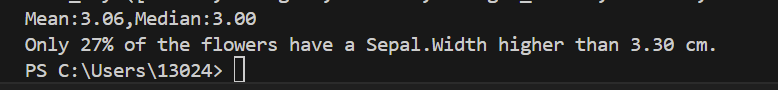
# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 1.d - Only 27% of the flowers have a Sepal.Width higher than \_\_\_\_\_\_\_\_ cm.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

threshold = iris\_df["sepal width (cm)"].quantile(1-0.27)

print(f"Only 27% of the flowers have a Sepal.Width higher than {threshold:.2f} cm.")



# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sns.pairplot(iris\_df, vars=iris.feature\_names, hue="species", diag\_kind="hist")

plt.suptitle("Scatter Plots of Iris variables")

plt.show()

"""

There are 6 pairs:

Sepal.Length vs Sepal.Width

Sepal.Length vs Petal.Length

Sepal.Length vs Petal.Width

Sepal.Width vs Petal.Length

Sepal.Width vs Petal.Width

Petal.Length vs Petal.Width

"""

A screenshot of a computer

AI-generated content may be incorrect.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

Strongest: Petal.Length vs Petal.Width (usually tight linear relationship)

Weakest: Sepal.Width vs Sepal.Length (often more scattered)

"""

corr=iris\_df.corr()

print(corr)

A black background with white text

AI-generated content may be incorrect.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#2. Using the PlantGrowth dataset...

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

print(PlantGrowth)        # View/Check datasets

bins = np.arange(3.3, PlantGrowth["weight"].max() + 0.3, 0.3)

plt.hist(PlantGrowth["weight"], bins=bins, edgecolor="orange")

plt.title("Histogram of Plant Weights")

plt.xlabel("Weight")

plt.ylabel("Count")

plt.show()

A screenshot of a computer

AI-generated content may be incorrect.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 2.b - Make boxplots of weight separated by group in a single graph.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

sns.boxplot(x="group", y="weight", data=PlantGrowth, palette="Set2")

plt.title("Boxplot of Plant Weights by Group")

plt.show()

A screenshot of a graph

AI-generated content may be incorrect.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

Look at box plot,

Find minimum value for “trt2” (bottom whisker)

Estimate how many “trt1” values are below this value

"""

min\_trt2 = PlantGrowth[PlantGrowth['group']=="trt2"]["weight"].min()

approx\_val = min\_trt2

print(f"Definately more than 50% as minimum value for trt2 is: {approx\_val:.2f}")



# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 2.d - Find the exact percentage of the "trt1" weights that are below the minimum "trt2" weight.

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

min\_trt2 = PlantGrowth[PlantGrowth['group']=="trt2"]["weight"].min()

approx\_val = min\_trt2

trt1\_weights = PlantGrowth[PlantGrowth['group']=="trt1"]["weight"]

below\_min = (trt1\_weights < min\_trt2).sum()

percent\_below = below\_min/len(trt1\_weights) \* 100

print(f"Percentage: {percent\_below:.2f}%")



# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# 2.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 (in R, try running ?heat.colors and/or check out https://www.r-bloggers.com/palettes-in-r/).

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

filtered = PlantGrowth[PlantGrowth["weight"] > 5.5]

sns.countplot(x='group', data=filtered, palette='Spectral')

plt.title("Barplot of Groups ( Weight > 5.5)")

plt.show()

A screenshot of a computer

AI-generated content may be incorrect.