```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.ticker import FuncFormatter
from matplotlib.patches import Patch
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
import statsmodels.api as sm


import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv(r'C:\Users\Mohamed Fawzi\Desktop\Iris Analysis\
01_iris.csv')

df.head()
```

```
{"columns":[{"name":"index","rawType":"int64","type":"integer"},
{"name":"sepal_length","rawType":"float64","type":"float"},
{"name":"sepal_width","rawType":"float64","type":"float"},
{"name":"petal_length","rawType":"float64","type":"float"},
{"name":"petal_width","rawType":"float64","type":"float"},
{"name":"species","rawType":"object","type":"string"}],"ref":"30b83fcd
-00a9-4aa1-8396-fdb458a30c5b","rows":
[["0","5.1","3.5","1.4","0.2","setosa"],
["1","4.9","3.0","1.4","0.2","setosa"],
["2","4.7","3.2","1.3","0.2","setosa"],
["3","4.6","3.1","1.5","0.2","setosa"],
["4","5.0","3.6","1.4","0.2","setosa"]],"shape":
{"columns":5,"rows":5}}
```

Inspecting Data:

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

```
# checking for null values
df.isnull().sum()
```

```
{"columns":[{"name":"index","rawType":"object","type":"string"},
{"name":"0","rawType":"int64","type":"integer"}],"ref":"d797e7d2-d6c0-
430a-8f5d-91d6d68aea6b","rows":[["sepal_length","0"],
["sepal_width","0"],["petal_length","0"],["petal_width","0"],
["species","0"]],"shape":{"columns":1,"rows":5}}
```

```
# checking for duplicates
df.duplicated().sum()
```

```
3
```

```
# dropping duplicates
df.drop_duplicates(inplace=True)
```

Exploratory Data Analysis (EDA):

- General Statistics Summary:

```
df.describe()
```

```
{"columns":[{"name":"index","rawType":"object","type":"string"},
{"name":"sepal_length","rawType":"float64","type":"float"},
{"name":"sepal_width","rawType":"float64","type":"float"},
{"name":"petal_length","rawType":"float64","type":"float"},
{"name":"petal_width","rawType":"float64","type":"float"}],"ref":"0446
d519-ee69-45dd-b251-8c43597c204b","rows":
[["count","147.0","147.0","147.0","147.0"],
["mean","5.856462585034014","3.05578231292517","3.780272108843538","1.
2088435374149662"],
["std","0.8290998607345104","0.4370087068034353","1.7591108999509792",
"0.7578742052400408"],["min","4.3","2.0","1.0","0.1"],
["25%","5.1","2.8","1.6","0.3"],["50%","5.8","3.0","4.4","1.3"],
["75%","6.4","3.3","5.1","1.8"],
["max","7.9","4.4","6.9","2.5"]],"shape":{"columns":4,"rows":8}}
```

- Species Specific Statistics:

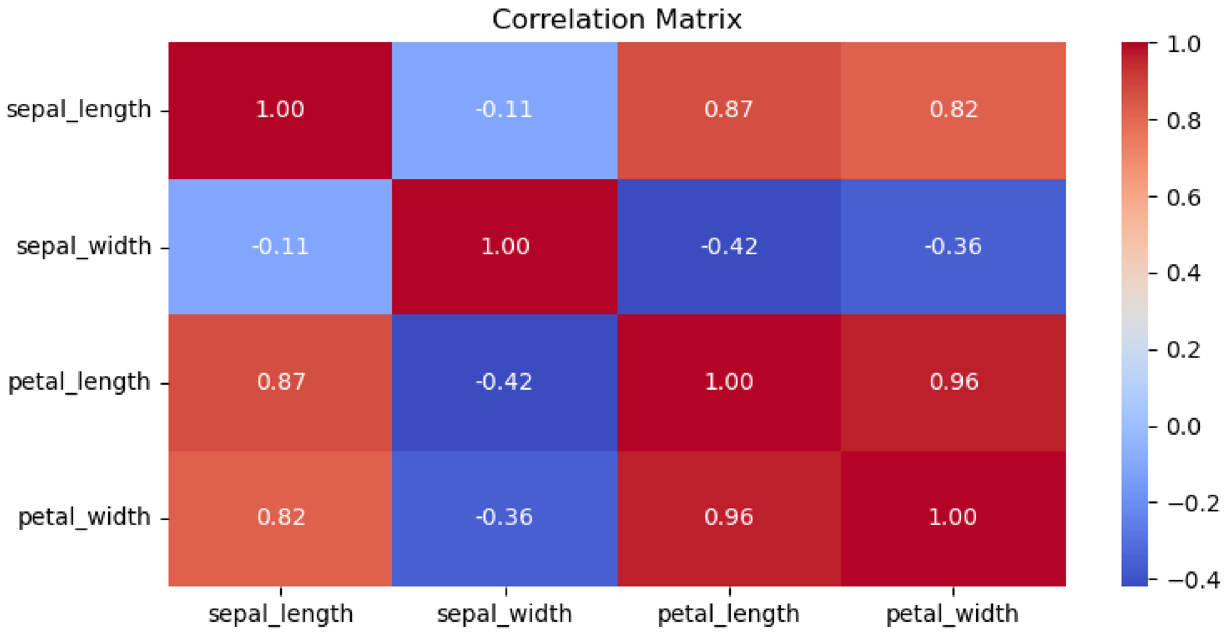
```
df.groupby('species').describe()
```

```
{"columns":[{"name":"species","rawType":"object","type":"string"},
{"name":"('sepal_length',
'count')","rawType":"float64","type":"float"},
{"name":"('sepal_length',
'mean')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'std')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'min')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'25%')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'50%')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'75%')","rawType":"float64","type":"float"},{"name":"('sepal_length',
'max')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'count')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'mean')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'std')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'min')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'25%')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'50%')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'75%')","rawType":"float64","type":"float"},{"name":"('sepal_width',
'max')","rawType":"float64","type":"float"},{"name":"('petal_length',
'count')","rawType":"float64","type":"float"},
{"name":"('petal_length',
'mean')","rawType":"float64","type":"float"},{"name":"('petal_length',
'std')","rawType":"float64","type":"float"},{"name":"('petal_length',
'min')","rawType":"float64","type":"float"},{"name":"('petal_length',
'25%')","rawType":"float64","type":"float"},{"name":"('petal_length',
'50%')","rawType":"float64","type":"float"},{"name":"('petal_length',
'75%')","rawType":"float64","type":"float"},{"name":"('petal_length',
'max')","rawType":"float64","type":"float"},{"name":"('petal_width',
'count')","rawType":"float64","type":"float"},{"name":"('petal_width',
'mean')","rawType":"float64","type":"float"},{"name":"('petal_width',
'std')","rawType":"float64","type":"float"},{"name":"('petal_width',
'min')","rawType":"float64","type":"float"},{"name":"('petal_width',
'25%')","rawType":"float64","type":"float"},{"name":"('petal_width',
'50%')","rawType":"float64","type":"float"},{"name":"('petal_width',
'75%')","rawType":"float64","type":"float"},{"name":"('petal_width',
'max')","rawType":"float64","type":"float"}],"ref":"2cd532d0-90a4-
4316-90c7-b0e353022d9c","rows":
[["setosa","48.0","5.010416666666667","0.35921876421948784","4.3","4.8
","5.0","5.2","5.8","48.0","3.43125","0.3832427429188974","2.3","3.2",
"3.4","3.7","4.4","48.0","1.4625000000000001","0.17700222381100905","1
.0","1.4","1.5","1.6","1.9","48.0","0.25","0.10518474122815553","0.1",
"0.2","0.2","0.3","0.6"],
["versicolor","50.0","5.936","0.5161711470638635","4.9","5.6","5.9","6
.3","7.0","50.0","2.7700000000000005","0.3137983233784114","2.0","2.52
5","2.8","3.0","3.4","50.0","4.26","0.46991097723995806","3.0","4.0","
4.35","4.6","5.1","50.0","1.3259999999999998","0.197752680004544","1.0
","1.2","1.3","1.5","1.8"],
["virginica","49.0","6.6040816326530605","0.6321125900744363","4.9","6
.3","6.5","6.9","7.9","49.0","2.979591836734694","0.32338031778692533"
,"2.2","2.8","3.0","3.2","3.8","49.0","5.5612244897959195","0.55370582
08601584","4.5","5.1","5.6","5.9","6.9","49.0","2.0285714285714285","0
.2768874620972691","1.4","1.8","2.0","2.3","2.5"]],"shape":
{"columns":32,"rows":3}}
```

Correlation Analysis:

- Let's visualize the correlation between the numerical features of the dataseet...

```
numeric_cols = ['sepal_length', 'sepal_width', 'petal_length',
'petal_width']

correlation_matrix = df[numeric_cols].corr()

plt.figure(figsize=(8, 4))
sns.heatmap(correlation_matrix,
            annot=True,
            cmap='coolwarm',
            fmt=".2f"
            )
plt.title('Correlation Matrix')

plt.tight_layout()
plt.show()
```



- **1. Strong Positive Correlations**

  - **Petal length and petal width:**
    Correlation = 0.96
    → Extremely strong positive relationship. As petal length increases, petal width also increases significantly.

  - **Sepal length and petal length:**
    Correlation = 0.87
    → Strong positive correlation. Longer sepals are associated with longer petals.

    - **Sepal length and petal width:**
      Correlation = 0.82
      → Strong positive relationship. Longer sepals correspond to wider petals.

- **2. Negative Correlations**

    - **Sepal width and petal length:**
      Correlation = -0.42
      → Moderate negative correlation. As petal length increases, sepal width tends to decrease.

    - **Sepal width and petal width:**
      Correlation = -0.36
      → Moderate negative correlation. Wider petals are associated with narrower sepals.

    - **Sepal length and sepal width:**
      Correlation = -0.11
      → Weak negative correlation. Sepal length and width are slightly inversely related, but this relationship is very weak.

- **3. Key Overall Insights**

    - Petal dimensions (length and width) are highly interrelated.

    - Sepal length is positively related to petal dimensions.

    - Sepal width tends to be inversely related to other features, especially petal size.

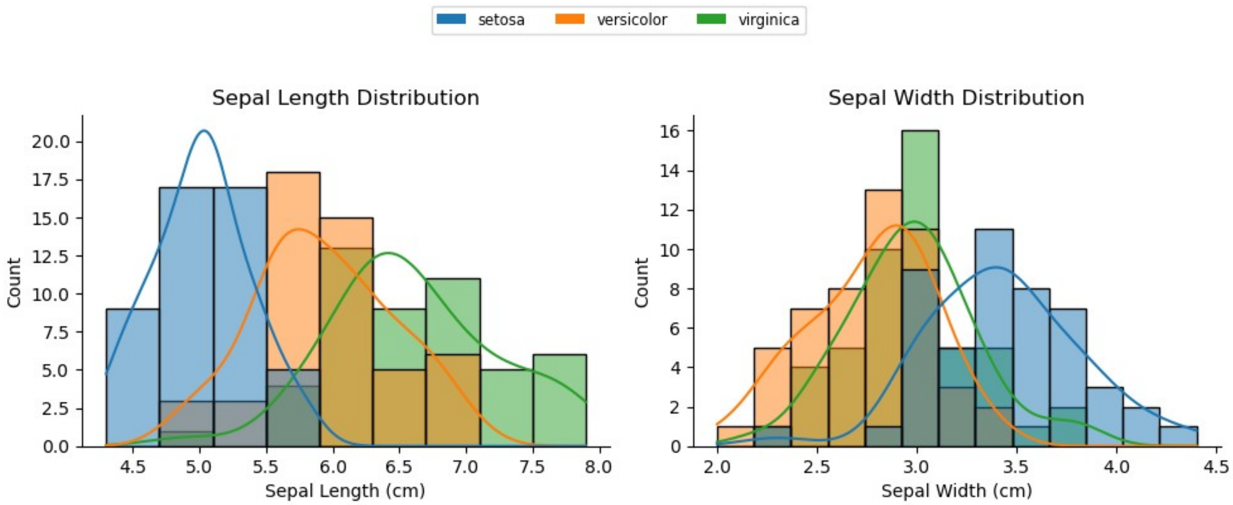
    - The strongest correlation in the dataset is between **petal length and petal width** (**0.96**).

Distribution Analysis

Sepal Length & Width Distribution

- Now, I will analyze the distribution of sepal measurements across different species to better nderstand thier variation.

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

palette = sns.color_palette('tab10')
species = df['species'].unique()

hist1 = sns.histplot(
    data=df,
    x='sepal_length',
    kde=True,
    ax=axes[0],
    hue='species',
    palette=palette
)
axes[0].set_title('Sepal Length Distribution')
axes[0].set_xlabel('Sepal Length (cm)')

axes[0].get_legend().remove()

hist2 = sns.histplot(
    data=df,
    x='sepal_width',
    kde=True,
    ax=axes[1],
    hue='species',
    palette=palette,
    legend=False
)
axes[1].set_title('Sepal Width Distribution')
axes[1].set_xlabel('Sepal Width (cm)')

handles = [Patch(facecolor=palette[i], label=species[i]) for i in
range(len(species))]

fig.legend(
    handles,
    [str(s) for s in species],
    loc='upper center',
    bbox_to_anchor=(0.5, 1.03),
    ncol=len(species),
    fontsize='small'
)

sns.despine()
plt.tight_layout()
plt.subplots_adjust(top=0.8)  # Make space for the legend
plt.show()
```



- **1. Sepal Length Distribution**
  - **Setosa:**
    → Sepal length is generally shorter, mostly between **4.5 cm and 5.5 cm**.
    → Distribution is tightly concentrated with a peak around **5.0 cm**.
  - **Versicolor:**
    → Sepal length ranges approximately from **5.0 cm to 7.0 cm**, with a concentration around **5.8 cm**.
    → Shows moderate spread.
  - **Virginica:**
    → Sepal length is the longest among species, ranging from **5.5 cm to 8.0 cm**, with a peak around **6.5 cm**.
    → Distribution is broader compared to Setosa and Versicolor.
- **Key Insight:**
  - Sepal length is a distinguishing feature among the species. Setosa is clearly separated from Virginica and Versicolor by having shorter sepals.
- **2. Sepal Width Distribution**
  - **Setosa:**
    → Sepal width is generally higher, mostly between **3.0 cm and 4.5 cm**.
    → Peak around **3.4 cm**, showing a broader and higher distribution compared to the other species.
  - **Versicolor:**
    → Sepal width is more centered around **2.8 cm**, with most values between **2.3 cm and 3.4 cm**.
    → Distribution is narrower and more symmetric.

  - **Virginica:**
    → Sepal width is spread between **2.2 cm and 3.8 cm**, peaking around **3.0 cm**.
    → Slightly more overlap with Versicolor.

- **Key Insight:**

  - Sepal width overlaps more among species than sepal length but Setosa tends to have wider sepals compared to Versicolor and Virginica.

- **3. Overall Observations**

  - **Sepal length** provides better separation between species, especially Setosa.

  - **Sepal width** has some overlapping regions, making it less discriminative but still useful.

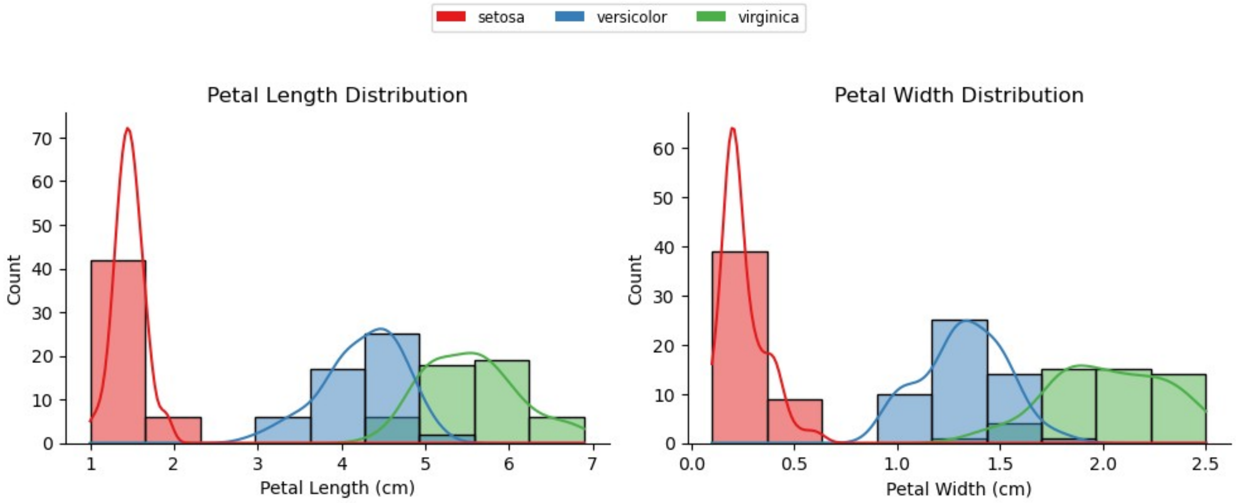
  - Kernel Density Estimate (KDE) plots complement histograms well by showing smooth probability distributions.

Petal Length & Width Distribution

- By plotting petal length and width side by side, this approach helps to identify distinguishing features among species, showcasing patterns and potential clustering.

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

palette = sns.color_palette('Set1')
species = df['species'].unique()

hist1 = sns.histplot(
    data=df,
    x='petal_length',
    kde=True,
    ax=axes[0],
    hue='species',
    palette=palette
)
axes[0].set_title('Petal Length Distribution')
axes[0].set_xlabel('Petal Length (cm)')

axes[0].get_legend().remove()

hist2 = sns.histplot(
    data=df,
    x='petal_width',
    kde=True,
    ax=axes[1],
    hue='species',
    palette=palette,
    legend=False
)
```

```
axes[1].set_title('Petal Width Distribution')
axes[1].set_xlabel('Petal Width (cm)')

handles = [Patch(facecolor=palette[i], label=species[i]) for i in
range(len(species))]

fig.legend(
    handles,
    [str(s) for s in species],
    loc='upper center',
    bbox_to_anchor=(0.5, 1.03),
    ncol=len(species),
    fontsize='small'
)

sns.despine()
plt.tight_layout()
plt.subplots_adjust(top=0.8)  # Make space for the legend
plt.show()
```



- **1. Petal Length Distribution**
  - **Setosa:**
    → Petal length is very short, tightly clustered between **1.0 cm and 2.0 cm**.
    → Sharp peak around **1.4 cm**, indicating very little variability.
  - **Versicolor:**
    → Petal length ranges from **3.0 cm to 5.0 cm**, peaking near **4.2 cm**.
    → Moderate spread with a relatively symmetric distribution.
  - **Virginica:**
    → Petal length ranges broadly from **4.5 cm to 7.0 cm**, peaking around **5.5**

cm.
→ Distribution is wider compared to Setosa and Versicolor.

- **Key Insight:**

  - Petal length clearly separates the species, especially **Setosa**, which does not overlap with Versicolor or Virginica.

- **2. Petal Width Distribution**

  - **Setosa:**
    → Petal width is very narrow, mostly between **0.1 cm and 0.6 cm**.
    → Sharp peak around **0.2 cm**, showing very low variation.

  - **Versicolor:**
    → Petal width ranges from **1.0 cm to 1.8 cm**, peaking near **1.3 cm**.
    → Moderate variability.

  - **Virginica:**
    → Petal width is the largest, spreading from **1.4 cm to 2.5 cm**, with a peak near **2.0 cm**.
    → Shows broader distribution compared to Versicolor.

- **Key Insight:**

  - Petal width, like petal length, is highly discriminative between species. **Setosa** is easily distinguished by its much smaller petal width.

- **3. Overall Observations**

  - Both **petal length** and **petal width** offer excellent separation among species.

  - **Setosa** is clearly distinct in both petal features with minimal to no overlap with other species.

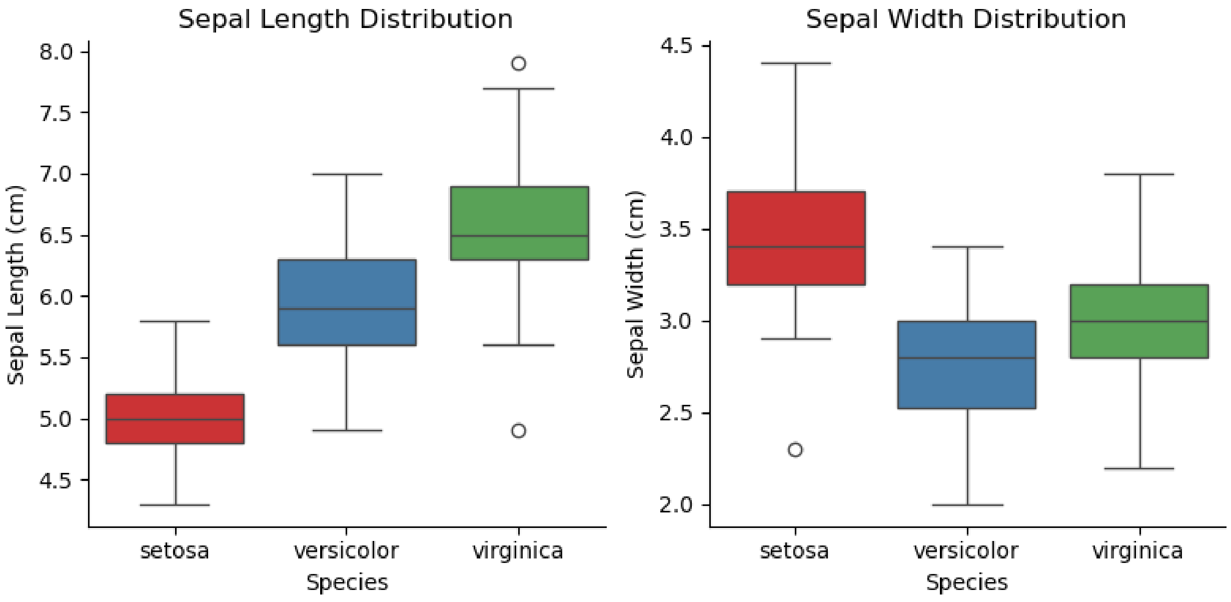
  - **Versicolor** and **Virginica** show some overlap but are still separable, especially in petal width.

Outliers

- Now, I will use boxplots to highlight **central tendency** and **spread** effectively.

Sepal Length & Width Outliers

```
fig, axes = plt.subplots(1, 2, figsize=(8, 4))
sns.boxplot(
    data=df,
    x='species',
    y='sepal_length',
    ax=axes[0],
    palette='Set1'
)
axes[0].set_title('Sepal Length Distribution')
axes[0].set_xlabel('Species')
axes[0].set_ylabel('Sepal Length (cm)')

sns.boxplot(
    data=df,
    x='species',
    y='sepal_width',
    ax=axes[1],
    palette='Set1'
)
axes[1].set_title('Sepal Width Distribution')
axes[1].set_xlabel('Species')
axes[1].set_ylabel('Sepal Width (cm)')


sns.despine()
plt.tight_layout()
plt.show()
```



- **1. Sepal Length Distribution**
  - **Setosa:**
    → Median around **5.0 cm** with a narrow interquartile range (IQR).
    → Lower sepal length compared to the other two species.
    → No significant outliers.
  - **Versicolor:**
    → Median around **5.9 cm** with a wider spread compared to Setosa.

→ Sepal length ranges from about **4.9 cm to 7.0 cm**.
→ One visible outlier above **7.0 cm**.

  - **Virginica:**
    → Median around **6.5 cm** with a broad distribution.
    → Sepal length ranges approximately from **5.0 cm to 8.0 cm**.
    → One lower outlier around **5.0 cm**.

- **Key Insight:**

  - Sepal length effectively differentiates the species, especially distinguishing Setosa from Versicolor and Virginica.

- **2. Sepal Width Distribution**

  - **Setosa:**
    → Median around **3.4 cm**, with the highest overall sepal width.
    → Wider spread compared to other species.
    → One lower outlier around **2.3 cm**.

  - **Versicolor:**
    → Median around **2.8 cm**, with a relatively tight spread.
    → Sepal width ranges from about **2.0 cm to 3.4 cm**.

  - **Virginica:**
    → Median around **3.0 cm**, showing moderate variability.
    → Sepal width overlaps with both Setosa and Versicolor, ranging from about **2.2 cm to 3.8 cm**.

- **Key Insight:**

  - Sepal width shows more **overlap** across species compared to sepal length, but Setosa generally has **higher sepal width** values.

- **3. Overall Observations**

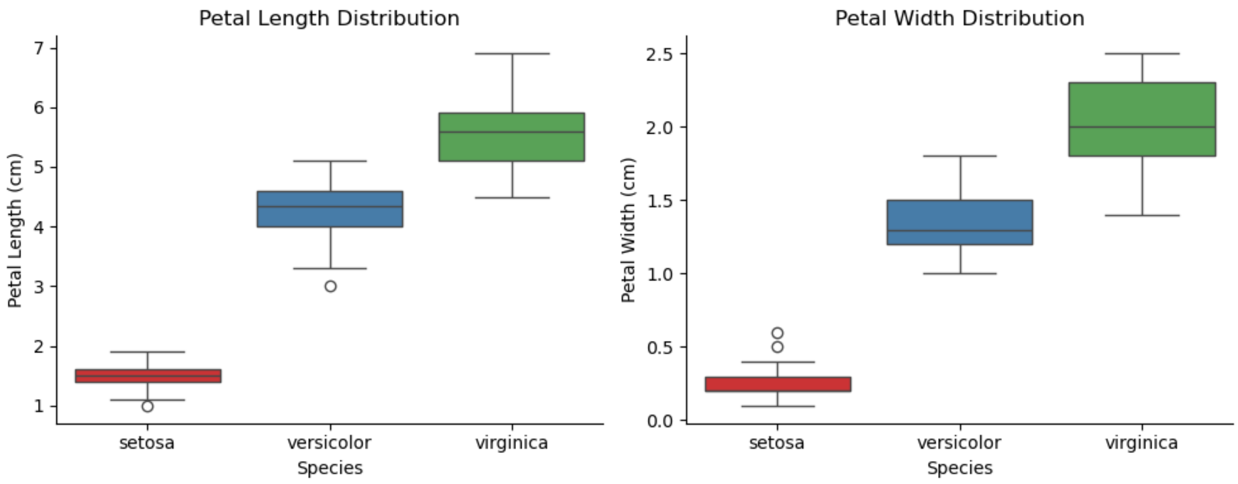
  - **Sepal length** is a more effective feature for species separation than sepal width.

  - Presence of some **outliers** indicates variability within species, especially in Versicolor and Virginica.

  - **Setosa** remains consistently distinguishable due to both shorter sepal length and wider sepal width.

Petal Length & Width Outliers

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4))
sns.boxplot(
    data=df,
    x='species',
    y='petal_length',
    ax=axes[0],
    palette='Set1'
)
axes[0].set_title('Petal Length Distribution')
axes[0].set_xlabel('Species')
axes[0].set_ylabel('Petal Length (cm)')

sns.boxplot(
    data=df,
    x='species',
    y='petal_width',
    ax=axes[1],
    palette='Set1'
)
axes[1].set_title('Petal Width Distribution')
axes[1].set_xlabel('Species')
axes[1].set_ylabel('Petal Width (cm)')


sns.despine()
plt.tight_layout()
plt.show()
#
```



- **1. Petal Length Distribution**
    - **Setosa:**
      → Petal length is consistently short, around **1.5 cm**, with minimal variation.
      → Range approximately from **1.0 cm to 2.0 cm**.
      → No significant outliers.
    - **Versicolor:**
      → Median around **4.3 cm**, with a range from **3.0 cm to 5.1 cm**.

    → Slight presence of outliers below **3.0 cm**.
    → Clear separation from Setosa.

  - **Virginica:**
    → Median around **5.5 cm**, with a wider spread from **4.5 cm to 7.0 cm**.
    → No outliers detected.

- **Key Insight:**

  - Petal length distinctly separates all three species with **no overlap** between Setosa and the other two species.

- **2. Petal Width Distribution**

  - **Setosa:**
    → Very narrow range, around **0.2 cm to 0.6 cm**.
    → Multiple outliers above **0.4 cm**, but overall very low variability.

  - **Versicolor:**
    → Median around **1.3 cm**, ranging from **1.0 cm to 1.8 cm**.
    → No significant outliers.

  - **Virginica:**
    → Median around **2.0 cm**, with a spread from **1.4 cm to 2.5 cm**.
    → Wider distribution compared to Versicolor.

- **Key Insight:**

  - Petal width, like petal length, shows excellent separation between species, with **Setosa** being clearly distinct.

- **3. Overall Observations**

  - **Petal measurements** are highly effective for differentiating between species.

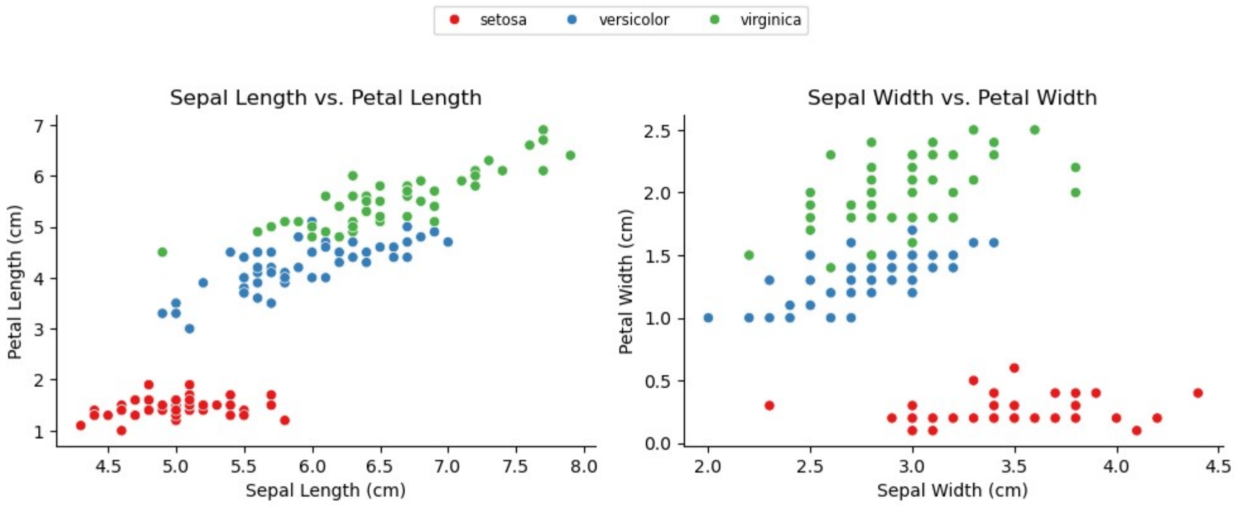
  - **Setosa** is distinctly separable based on both petal length and width.

  - Some **overlap** exists between **Versicolor** and **Virginica**, but ranges are still largely distinct.

  - Presence of **minor outliers** mainly in Setosa's petal width.

Scatter Plot Matrix

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

scatter1 = sns.scatterplot(
    data=df,
    x='sepal_length',
    y='petal_length',
    hue='species',
    palette='Set1',
    ax=axes[0]
)
axes[0].set_title('Sepal Length vs. Petal Length')
axes[0].set_xlabel('Sepal Length (cm)')
axes[0].set_ylabel('Petal Length (cm)')

handles, labels = scatter1.get_legend_handles_labels()

axes[0].get_legend().remove()

scatter2 = sns.scatterplot(
    data=df,
    x='sepal_width',
    y='petal_width',
    hue='species',
    palette='Set1',
    ax=axes[1],
    legend=False
)
axes[1].set_title('Sepal Width vs. Petal Width')
axes[1].set_xlabel('Sepal Width (cm)')
axes[1].set_ylabel('Petal Width (cm)')

fig.legend(
    handles,
    labels,
    loc='upper center',
    bbox_to_anchor=(0.5, 1.03),
    ncol=len(labels),
    fontsize='small'
)

sns.despine()

plt.tight_layout()
plt.subplots_adjust(top=0.8)
plt.show()
```



- **1. Sepal Length vs. Petal Length**
  - **Setosa (Red):**
    → Clustered distinctly in the **lower-left corner** with short sepal and petal lengths.
    → Sepal length ranges from **4.5 cm to ~5.8 cm**.
    → Petal length remains consistently low (**~1.0 cm to 2.0 cm**).
    → Very **low variation**.
  - **Versicolor (Blue):**
    → Forms a **moderate cluster** with a **positive linear trend** between sepal and petal lengths.
    → Sepal length ranges from **~5.0 cm to ~7.0 cm**.
    → Petal length varies from **~3.0 cm to ~5.0 cm**.
  - **Virginica (Green):**
    → **Clearly separated** from Setosa and overlaps slightly with Versicolor at lower values.
    → Sepal length ranges from **~5.5 cm to ~8.0 cm**.
    → Petal length is the **longest**, ranging from **~4.5 cm to ~7.0 cm**.
    → Displays a **strong positive correlation**.
- **Key Insight:**
  - There is a **strong positive relationship** between **sepal length** and **petal length**, especially for **Versicolor** and **Virginica**.
    Setosa is **clearly distinct** with **low values** for both features.
- **2. Sepal Width vs. Petal Width**
  - **Setosa (Red):**
    → Distinct **low petal width** values between **~0.1 cm and ~0.6 cm**.
    → Sepal width is comparatively **higher** than petal width (**~2.5 cm to ~4.5**

cm).
→ **No overlap** with other species in petal width.

  - **Versicolor (Blue):**
    → Sepal width varies from **~2.0 cm to ~3.5 cm**, while petal width ranges from **~1.0 cm to ~1.8 cm**.
    → Shows a **moderate positive trend**.

  - **Virginica (Green):**
    → Has the **highest petal width**, ranging from **~1.5 cm to ~2.5 cm**.
    → Sepal width is moderately spread from **~2.5 cm to ~3.8 cm**.

- **Key Insight:**

  - Both **Versicolor** and **Virginica** show **overlap** in sepal width but are more **distinct in petal width**.
    Setosa is **clearly isolated** with **small petal width** values.

- **3. General Observations**

  - **Setosa** species are **well-separated** from the others based on petal dimensions.

  - **Versicolor** and **Virginica** show **gradual increases** in both petal and sepal measurements, though **some overlap exists**.

  - **Positive correlation** is evident in both scatter plots, particularly for petal dimensions.

Selecting Features for Regression Analysis

- Predection the petal length using the sepal_length and witdth

```
X = df[['sepal_length', 'sepal_width']]  #features
y = df['petal_length']                   #target

#splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, 
test_size=0.2, random_state=42)

#using scukit-learn's LinearRegression model for prediction
model = LinearRegression()
model.fit(X_train, y_train)

print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

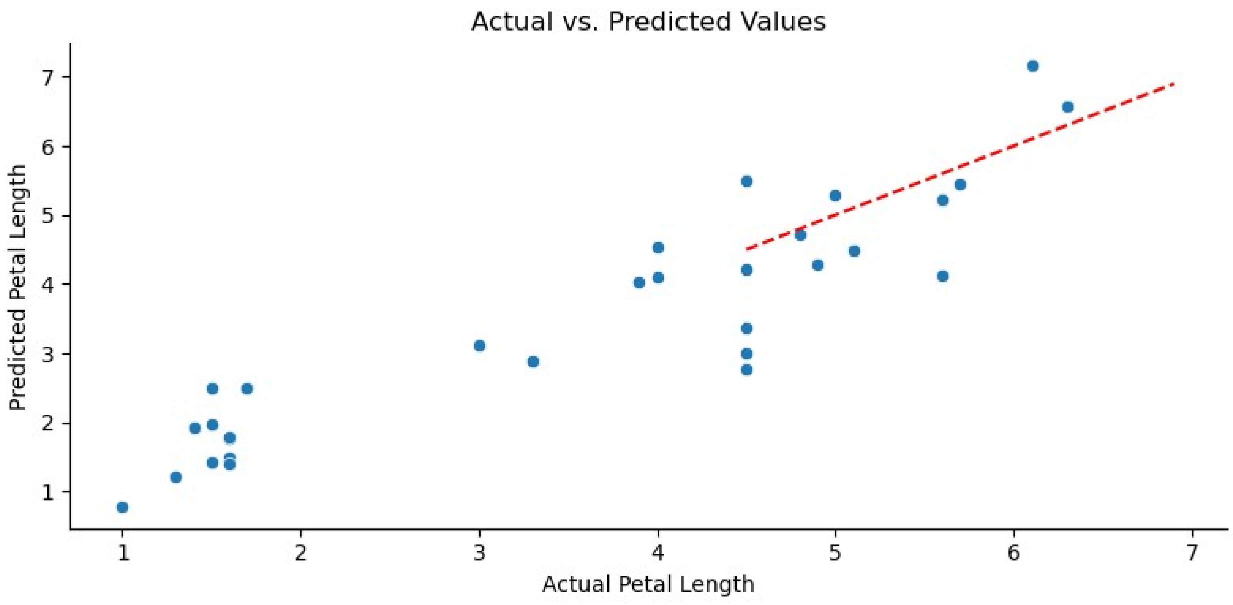
```
Coefficients: [ 1.80438018 -1.30611191]
Intercept: -2.8123264432381676
```

```
#using statsmodels for detailed regression analysis
X_train_sm = sm.add_constant(X_train)
```

```
sm_model = sm.OLS(y_train, X_train_sm).fit()

sm_model.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results                            

======================================================================
========
Dep. Variable:           petal_length   R-squared:                  
0.874
Model:                            OLS   Adj. R-squared:             
0.872
Method:                 Least Squares   F-statistic:                
395.6
Date:                ter, 27 mai 2025   Prob (F-statistic):         
5.13e-52
Time:                        16:52:34   Log-Likelihood:             
-110.84
No. Observations:                 117   AIC:                        
227.7
Df Residuals:                     114   BIC:                        
236.0
Df Model:                           2                               

Covariance Type:            nonrobust                                

======================================================================
==========
                   coef    std err          t      P>|t|      [0.025  
0.975]
----------------------------------------------------------------------
----------
const           -2.8123      0.628     -4.478      0.000      -4.056  
-1.568
sepal_length     1.8044      0.072     25.070      0.000       1.662  
1.947
sepal_width     -1.3061      0.134     -9.761      0.000      -1.571  
-1.041
======================================================================
========
Omnibus:                        2.441   Durbin-Watson:              
2.150
Prob(Omnibus):                  0.295   Jarque-Bera (JB):           
2.286
Skew:                           0.261   Prob(JB):                   
0.319
Kurtosis:                       2.556   Cond. No.                   
74.0
```

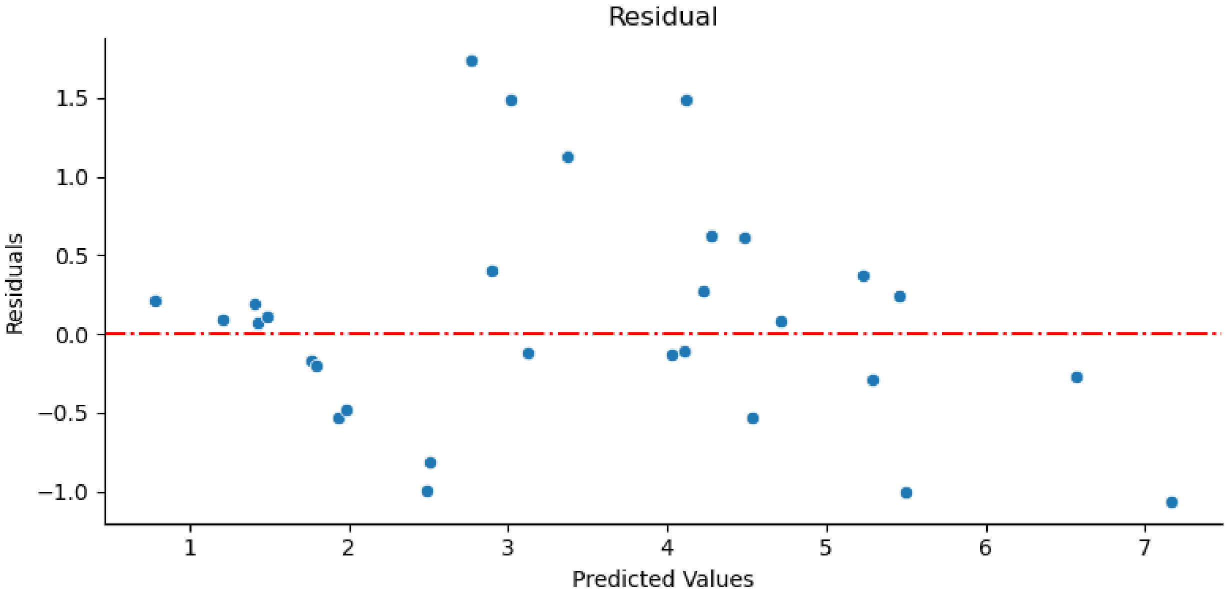
```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""

y_pred = model.predict(X_test)

print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
print("R-squared:", r2_score(y_test, y_pred))

Mean Squared Error: 0.4952016809380731
R-squared: 0.8308153191773624

plt.figure(figsize=(8, 4))
sns.scatterplot(
    x=y_test,
    y=y_pred
    )

plt.xlabel("Actual Petal Length")
plt.ylabel("Predicted Petal Length")
plt.title("Actual vs. Predicted Values")
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--')  #prediction line


sns.despine()
plt.tight_layout()
plt.show()
```



- **Model Performance: Actual vs. Predicted Petal Length**
    - **Plot Description**
        - **X-axis:** Actual Petal Length (cm)
        - **Y-axis:** Predicted Petal Length (cm)
        - **Blue Dots:** Represent individual **data points** (actual vs. predicted).
        - **Red Dashed Line:** Ideal **perfect fit** line where predicted = actual.
    - **Key Observations**
        - Most points are **closely clustered** around the red dashed line, indicating that the model's predictions are **reasonably accurate**.
        - A few **outliers** are noticeable, particularly for **mid-range** and **higher petal lengths** — this suggests that the model may slightly **underfit** or **overfit** some specific regions.
        - The overall trend suggests a **positive correlation** between **actual** and **predicted** values.
    - **Model Evaluation Insights**
        - The red dashed line represents the **ideal scenario**: predicted values should match the actual values.
        - Deviations from this line reflect **prediction errors**.
        - The plot suggests that the model **performs well**, but improvements may be possible through:
            - **Feature engineering**.
            - **Model complexity tuning**.
            - Addressing potential **heteroscedasticity** (increased spread at certain ranges).

```
residuals = y_test - y_pred
plt.figure(figsize=(8, 4))
sns.scatterplot(
    x=y_pred,
    y=residuals
    )
plt.axhline(y=0, color='r', linestyle='-.')

plt.xlabel("Predicted Values")
plt.ylabel("Residuals")
plt.title("Residual")


sns.despine()
```

```
plt.tight_layout()
plt.show()
```



- **Residual Plot: Predicted vs. Residuals**
- **Plot Description**
    - **X-axis:** Predicted Petal Length (cm)
    - **Y-axis:** Residuals (Actual - Predicted)
    - **Blue Dots:** Represent **residuals** for each prediction.
    - **Red Dashed Line: Zero residual line** indicating perfect predictions.
- **Key Observations**
    - The residuals appear to be **randomly scattered** around the zero line, which is generally a good sign indicating that the model does **not exhibit clear bias**.
    - Some **outliers** and points with relatively **high residuals** are visible, particularly at **lower** and **mid-range predicted values**.
    - No obvious **non-linear patterns** are present, suggesting that the model's assumptions of **linearity** and **homoscedasticity** (constant variance) may be **reasonably satisfied**.
- **Model Diagnostics**
    - **Homoscedasticity:**
      The spread of residuals seems to be **fairly constant** across predicted values, though some variability increases slightly for certain ranges.

  - **Independence:**
    No clear trends or systematic patterns in residuals, indicating **independence** of errors.

  - **Normality:**
    Additional diagnostics like a **Q-Q plot** would be required to confirm **normal distribution** of residuals.

- **Conclusion**

The residual plot indicates that the model performs **reasonably well** with **no major violations** of regression assumptions. However, some **outliers** may merit closer inspection for potential data issues or model refinements.

```python
for species in df['species'].unique():
    species_data = df[df['species'] == species]
    X = species_data[['sepal_length', 'sepal_width']]
    y = species_data['petal_length']

    model = LinearRegression().fit(X, y)
    print(f"\nRegression for {species}:")
    print("R-squared:", model.score(X, y))
```

```
Regression for setosa:
R-squared: 0.07183204228611262

Regression for versicolor:
R-squared: 0.6057506078304831

Regression for virginica:
R-squared: 0.7447548953125704
```

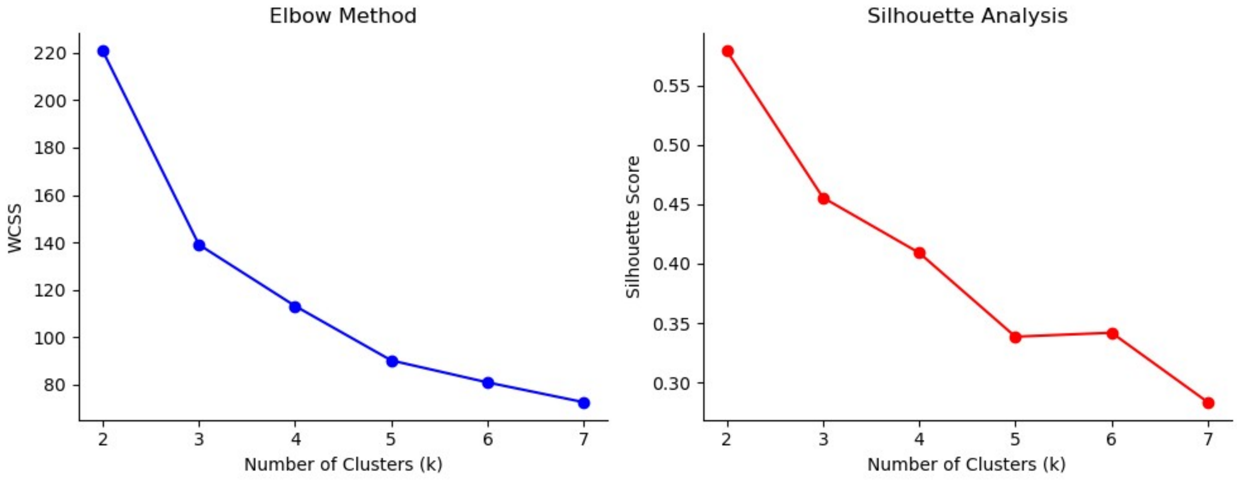
Cluster Analysis (K-Means):

- Preprocess Data

```python
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score

# Select features (exclude species label for unsupervised learning)
X = df.drop('species', axis=1)

# Standardize features (K-Means is sensitive to scale)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

- Determine Optimal Number of Clusters (Elbow Method):

```
wcss = []  #within cluster sum of squares
silhouette_scores = []
k_range = range(2, 8)

for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)
    silhouette_scores.append(silhouette_score(X_scaled,
kmeans.labels_))

# Plot Elbow Curve
plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1)
plt.plot(k_range, wcss, 'bo-')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('WCSS')
plt.title('Elbow Method')


plt.subplot(1, 2, 2)
plt.plot(k_range, silhouette_scores, 'ro-')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Analysis')


sns.despine()
plt.tight_layout()
plt.show()
```



Clustering Evaluation: Elbow Method and Silhouette Analysis

- **1. Elbow Method**

- **Plot Description:**
    - **X-axis:** Number of Clusters (k)
    - **Y-axis:** WCSS (Within-Cluster Sum of Squares)
    - **Blue Line:** Shows the reduction in WCSS as k increases.

- **Key Observation:**
  - The **elbow point** appears around **k = 3**.
  - After k = 3, the WCSS reduction slows, indicating **diminishing returns** in reducing within-cluster variance.

- **Interpretation:**
  - **Optimal number of clusters:**
    - Likely between **3 and 4**, where adding more clusters doesn't significantly improve compactness.

- **2. Silhouette Analysis**
  - **Plot Description:**
    - **X-axis:** Number of Clusters (k)
    - **Y-axis:** Silhouette Score
    - **Red Line:** Indicates the average silhouette score for different k.

- **Key Observation:**
  - The **highest silhouette score** occurs at **k = 2**, but a reasonable balance occurs around **k = 3**.
  - Scores steadily **decrease** as k increases beyond 3.

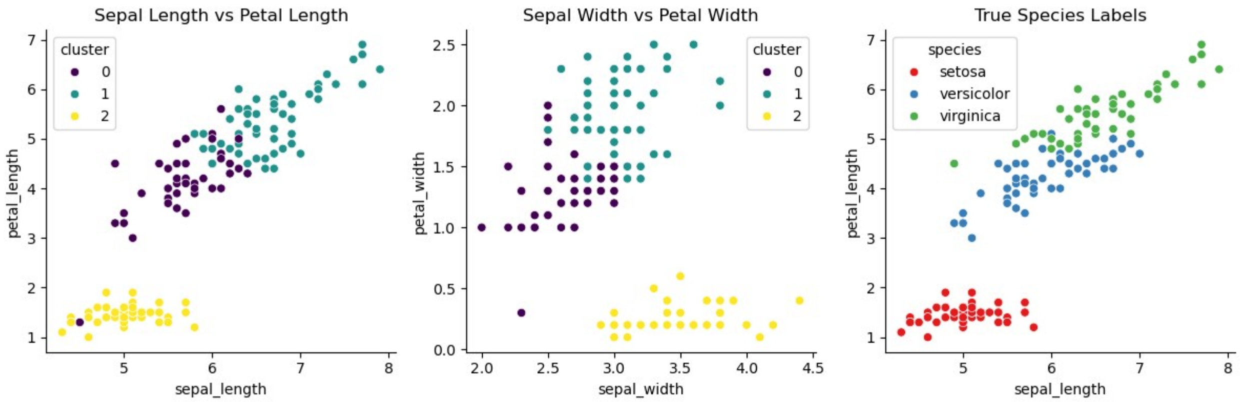
- **Interpretation:**
  - **k = 2** gives the highest cohesion and separation, but **k = 3** might offer a more **meaningful segmentation** of data with acceptable cohesion.
  - Clustering quality **declines** beyond **k = 3 or 4**, as shown by decreasing silhouette scores.

- **Overall Conclusion:**
  - **Recommended k:** Between **2 and 3** based on both **Elbow** and **Silhouette** methods.
  - Consider **domain knowledge** and **interpretability** when selecting the final number of clusters.

- Apply K-Means with Optimal k

```
k = 3  #based on elbow/silhouette (matches true species count)
kmeans = KMeans(n_clusters=k, random_state=42)
clusters = kmeans.fit_predict(X_scaled)
```

```python
# Add clusters to original DataFrame
df['cluster'] = clusters

plt.figure(figsize=(12, 4))

# Sepal Length vs Petal Length
plt.subplot(1, 3, 1)
sns.scatterplot(
    data=df,
    x='sepal_length',
    y='petal_length',
    hue='cluster',
    palette='viridis'
    )

plt.title('Sepal Length vs Petal Length')

#Sepal Width vs Petal Width
plt.subplot(1, 3, 2)
sns.scatterplot(
    data=df,
    x='sepal_width',
    y='petal_width',
    hue='cluster',
    palette='viridis'
    )

plt.title('Sepal Width vs Petal Width')

# True Species Labels (for comparison)
plt.subplot(1, 3, 3)
sns.scatterplot(
    data=df,
    x='sepal_length',
    y='petal_length',
    hue='species',
    palette='Set1'
    )

plt.title('True Species Labels')


sns.despine()
plt.tight_layout()
plt.show()
```



Clustering vs True Labels: Iris Dataset

- **1. Cluster Assignments**
  - **Plots** show the **k-means clustering** result with **k = 3**:
    - **Left:** Sepal Length vs Petal Length
    - **Center:** Sepal Width vs Petal Width
    - **Right:** True species labels
  - **Clusters:**
    - **Cluster 0:** Purple
    - **Cluster 1:** Teal
    - **Cluster 2:** Yellow
- **Key Observations:**
  - **Setosa (red points)** is **clearly separated** as **Cluster 2** (yellow) — indicating **perfect clustering** for this class.
  - **Versicolor (blue)** and **Virginica (green)** are **more overlapping**, primarily split into **Clusters 0 and 1**:
  - Shows **partial overlap**, consistent with known biological characteristics.
  - Clustering aligns well with the **natural groupings** in the dataset.
- **2. Interpretation:**
  - **Setosa:** Easily distinguishable with distinct morphological traits.
  - **Versicolor vs Virginica:** More subtle separation; clustering captures much but not all of this.
  - K-means clustering performs **well** but has some **misclassifications** likely due to the **linear boundary assumption** of k-means.
- **Summary:**

  - K-means effectively identifies **three clusters** that broadly correspond to the **three Iris species**.
  - Minor overlap in **Versicolor** and **Virginica** highlights the **challenge of perfect clustering** in this dataset.

- ### Overall Conclusion: Iris Dataset Analysis

  a. **Exploratory Data Analysis (EDA)**
  - **Boxplots** showed clear differences among species for **sepal** and **petal** measurements.
  - **Petal features** (length and width) offer better separation across species compared to sepal features.
  - **Setosa** is distinct with small petal dimensions, while **Versicolor** and **Virginica** overlap more but still show progressive increases in size.

  b. **Correlation Analysis**
  - Strong **positive correlations** between **sepal length and petal length** as well as **sepal width and petal width**.
  - This suggests coordinated growth patterns within flower structures.

  c. **Regression Modeling**
  - Regression analysis relating **sepal length** to **petal length** shows a generally linear relationship.
  - **Residual plots** indicate some **heteroscedasticity** but no major anomalies.
  - Model performs well, but minor deviations suggest a simple linear model may not capture all complexities.

  d. **Clustering and Validation**
  - **Elbow method** and **silhouette scores** indicated that **k = 3** is an appropriate number of clusters, aligning with the three Iris species.
  - **K-means clustering** successfully identified three distinct groups:
    - **Setosa** was perfectly clustered.
    - Some **overlap** between **Versicolor** and **Virginica**, reflecting the biological similarity between these species.
  - **Cluster visualization** confirmed these patterns and validated the suitability of k-means.

  e. **General Insights**
  - The Iris dataset's **structure** is well captured by both **supervised** and **unsupervised** learning methods.
  - **Setosa** is consistently the easiest species to classify.
  - **Versicolor** and **Virginica** present challenges due to feature **overlaps**, suggesting potential benefits from more **advanced models** or **non-linear decision boundaries**.

- Prepared by: Mohamed Fawzi.
- Email : modyreturn@gmail.com
- Github : https://github.com/modyreturn

- Linkedin : www.linkedin.com/in/mofawzirj