

data set :iris link: <https://www.kaggle.com/datasets/uciml/iris>

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('/content/Iris.csv')
df.head()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

```
print("Dataset Info:")
df.info()

print("\nSummary Statistics:")
df.describe()
```

Dataset Info:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Id                   150 non-null   int64
1   SepalLengthCm       150 non-null   float64
2   SepalWidthCm        150 non-null   float64
3   PetalLengthCm       150 non-null   float64
4   PetalWidthCm        150 non-null   float64
5   Species              150 non-null   object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

Summary Statistics:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

Start coding or [generate](#) with AI.

```
print("Missing Values per Column:")
print(df.isnull().sum())

num_cols = df.select_dtypes(include=['float64', 'int64']).columns
cat_cols = df.select_dtypes(exclude=['float64', 'int64']).columns

for col in num_cols:
    df[col].fillna(df[col].median(), inplace=True)

for col in cat_cols:
    df[col].fillna(df[col].mode()[0], inplace=True)

print("\nMissing Values After Cleaning:")
print(df.isnull().sum())
```

Missing Values per Column:

```
Id          0
SepalLengthCm  0
SepalWidthCm  0
PetalLengthCm  0
PetalWidthCm  0
Species      0
dtype: int64
```

Missing Values After Cleaning:

```
Id          0
SepalLengthCm  0
SepalWidthCm  0
PetalLengthCm  0
PetalWidthCm  0
Species      0
dtype: int64
```

/tmp/ipython-input-1003229514.py:8: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment. The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col]

```
df[col].fillna(df[col].median(), inplace=True)
```

/tmp/ipython-input-1003229514.py:11: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment. The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col]

```
df[col].fillna(df[col].mode()[0], inplace=True)
```

```
def handle_outliers(df, col):
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    df[col] = np.where(df[col] < lower, lower, np.where(df[col] > upper, upper, df[col]))
```

```
num_cols = df.select_dtypes(include=['float64', 'int64']).columns
```

```
for col in num_cols:
    handle_outliers(df, col)
```

```
print("Outliers handled successfully.")
```

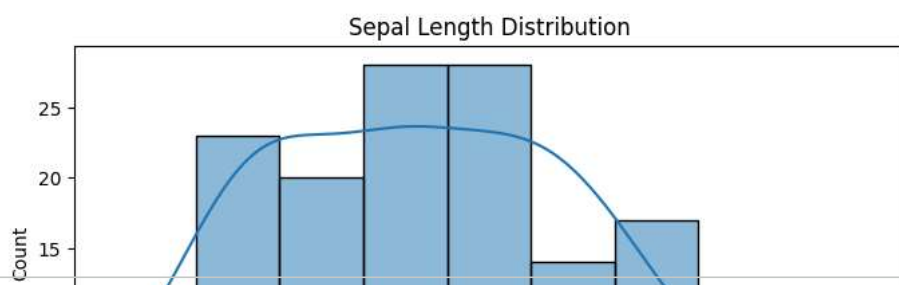
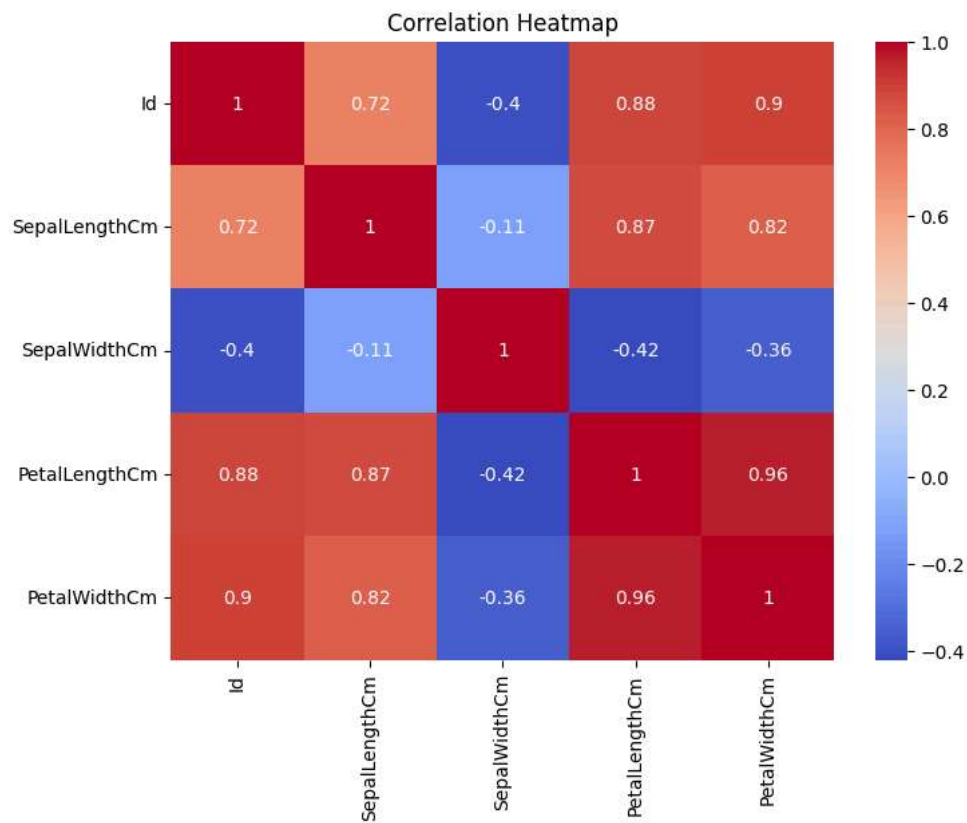
Outliers handled successfully.

```
# Correlation heatmap
plt.figure(figsize=(8,6))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

```
# Distribution of sepal length
plt.figure(figsize=(8,4))
sns.histplot(df['SepalLengthCm'], kde=True)
plt.title('Sepal Length Distribution')
plt.show()
```

```
# Boxplot for petal width
plt.figure(figsize=(8,4))
sns.boxplot(x=df['PetalWidthCm'])
plt.title('Petal Width Boxplot')
plt.show()
```

```
# Countplot for species
plt.figure(figsize=(6,4))
sns.countplot(x='Species', data=df, palette='pastel')
plt.title('Species Distribution')
plt.show()
```



```
df.to_csv('/content/cleaned_iris_dataset.csv', index=False)
print("Cleaned dataset saved successfully.")
```

