Data Mining — Task 1: Data Preprocessing and Exploration

1. Dataset Description

For this task, the Iris dataset from scikit-learn was used. It contains 150 samples across three species of Iris (setosa, versicolor, and virginica) with four numeric features:

- Sepal Length (cm)
- Sepal Width (cm)
- Petal Length (cm)
- Petal Width (cm)

The target variable is the species of the flower. The dataset is well-balanced with 50 samples per class.

	sepal_length	sepal_width	petal_length	petal_width	species_0	species_1	species_2
2	0.2222222222222221	0.625	0.06779661016949151	0.04166666666666667	1.0	0.0	0.0
3	0.16666666666666674	0.41666666666666674	0.06779661016949151	0.04166666666666667	1.0	0.0	0.0
4	0.11111111111111116	0.5	0.05084745762711865	0.04166666666666667	1.0	0.0	0.0
5	0.08333333333333326	0.45833333333333326	0.0847457627118644	0.04166666666666667	1.0	0.0	0.0

2. Data Loading

The dataset was loaded into a Pandas DataFrame using the following approach:

```
from sklearn.datasets import load_iris
import pandas as pd

iris = load_iris(as_frame=True)
df = iris.frame
```

The dataset shape was (150, 5) with no missing values detected.

3. Preprocessing Steps

- Missing Values Check:

- Used df.isnull().sum() to confirm there were no missing entries in any column.

```
Missing values per column:
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
```

- **Feature Scaling:**
  - Applied Min-Max Scaling (range 0–1) to numerical features using MinMaxScaler from sklearn.preprocessing.
  - This ensures all features contribute equally to distance-based algorithms such as K-Means.
- **Label Encoding:**

The target species column was converted into one-hot encoded columns (species_0, species_1, species_2) to make it suitable for certain machine learning models.

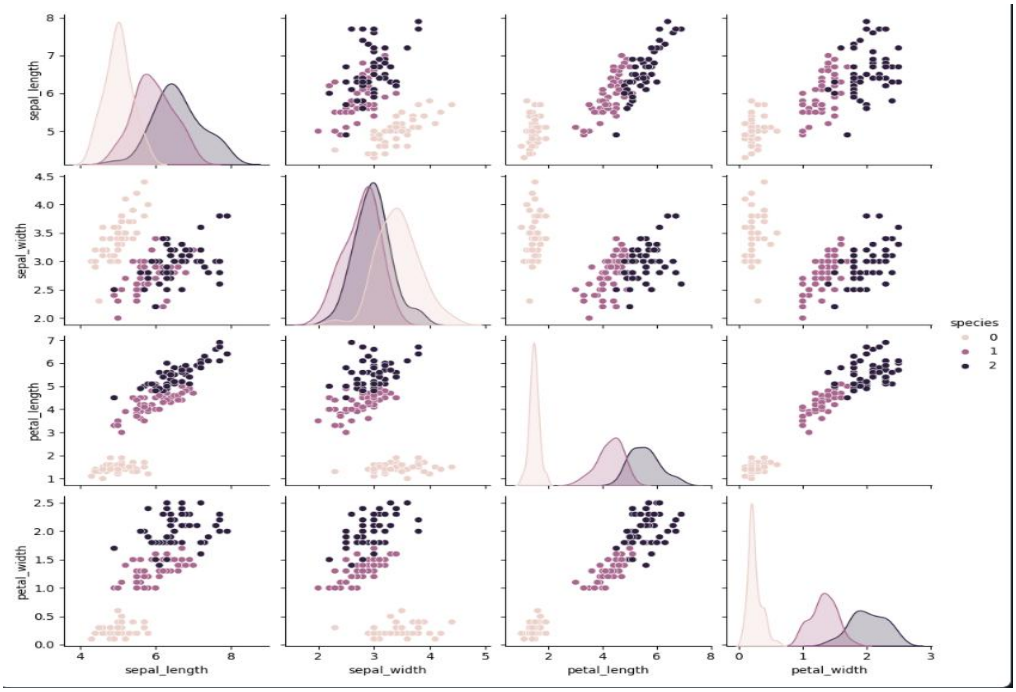
4. Data Exploration

- Summary Statistics:
  - Computed using df.describe().
  - The mean, standard deviation, min, and max values indicated that petal length and petal width have the highest variation across species, suggesting these features are strong differentiators.

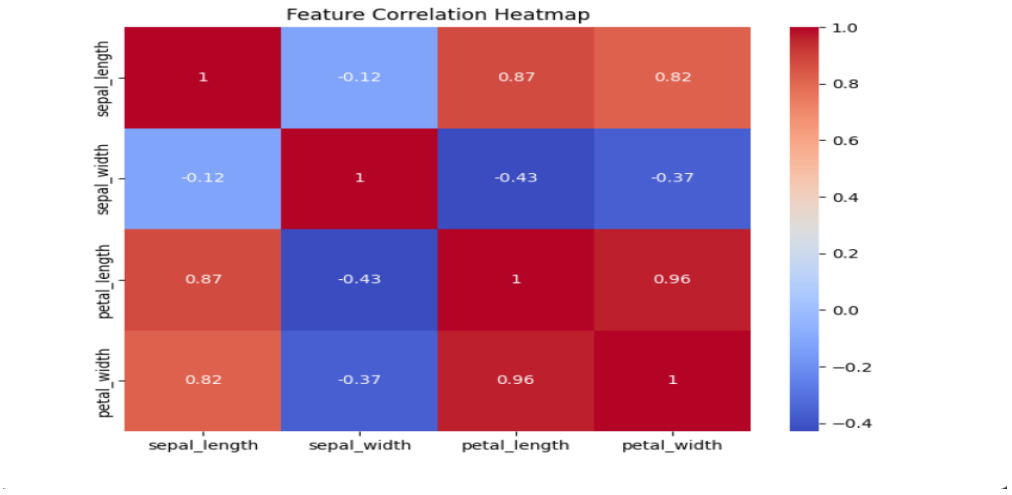
```
Summary statistics:
       sepal_length  sepal_width  petal_length  petal_width     species
count    150.000000   150.000000    150.000000   150.000000  150.000000
mean       5.843333     3.057333      3.758000     1.199333    1.000000
std        0.828066     0.435866      1.765298     0.762238    0.819232
min        4.300000     2.000000      1.000000     0.100000    0.000000
25%        5.100000     2.800000      1.600000     0.300000    0.000000
50%        5.800000     3.000000      4.350000     1.300000    1.000000
75%        6.400000     3.300000      5.100000     1.800000    2.000000
max        7.900000     4.400000      6.900000     2.500000    2.000000
```

- **Pairplot Visualization:**

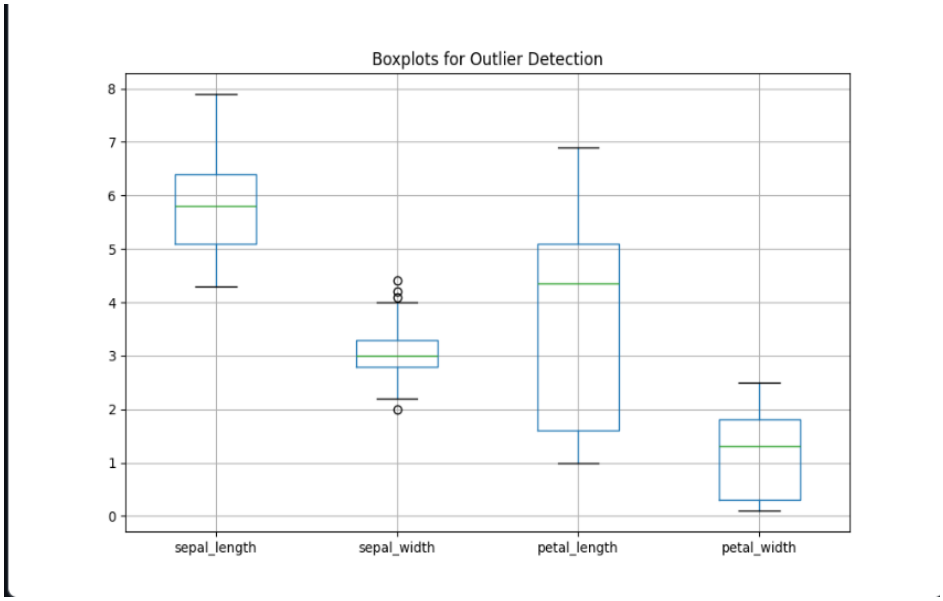
- A Seaborn pairplot showed clear separation between species in the petal measurements, while sepal measurements overlapped more.



- **Correlation Heatmap:**
  - Strong positive correlation was observed between petal length and petal width (r ≈ 0.96).
  - Sepal length also had moderate positive correlation with petal length and width.



- **Outlier Detection:**
    - Boxplots for each feature revealed a few mild outliers in sepal width, but none were extreme enough to require removal.



5. Train/Test Split Function

- A reusable function was created to split the dataset into training (80%) and testing (20%) sets:

```
from sklearn.model_selection import train_test_split

def split_data(X, y, test_size=0.2, random_state=42):
    return train_test_split(X, y, test_size=test_size, random_state=random
```

This ensures reproducible splits for downstream modeling.

```
Train shape: (120, 4), Test shape: (30, 4)
Processed dataset saved to iris_processed.csv
```

6. Conclusion

The preprocessing stage produced a scaled, clean, and encoded dataset ready for machine learning. Exploration highlighted that petal measurements are more distinctive for species classification than sepal measurements. The dataset's balanced nature and lack of missing data reduce preprocessing complexity, allowing focus on model experimentation in subsequent tasks.