A_1.py\n==========

```
 1. Read CSV file
CSV file not found, skipping...

 2. Read Excel file
Excel file not found, skipping...

 3. Read JSON file
JSON file not found, skipping...

 4. Read CSV from URL
  Month   "1958"   "1959"   "1960"
0   JAN      340      360      417
1   FEB      318      342      391
2   MAR      362      406      419
3   APR      348      396      461
4   MAY      363      420      472

 5. Built-in dataset from sklearn
   sepal length (cm)  sepal width (cm)  ...  petal width (cm)  target
0                5.1               3.5  ...               0.2       0
1                4.9               3.0  ...               0.2       0
2                4.7               3.2  ...               0.2       0
3                4.6               3.1  ...               0.2       0
4                5.0               3.6  ...               0.2       0

[5 rows x 5 columns]

 6. Built-in dataset from seaborn
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4

 7. Display dataset info and description

--- Iris Dataset 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 (cm)  150 non-null    float64
```

```
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   target             150 non-null    int64
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
None

--- Iris Dataset Description ---
       sepal length (cm)  sepal width (cm)  ...  petal width (cm)
target
count         150.000000        150.000000  ...        150.000000
150.000000
mean            5.843333          3.057333  ...          1.199333
1.000000
std             0.828066          0.435866  ...          0.762238
0.819232
min             4.300000          2.000000  ...          0.100000
0.000000
25%             5.100000          2.800000  ...          0.300000
0.000000
50%             5.800000          3.000000  ...          1.300000
1.000000
75%             6.400000          3.300000  ...          1.800000
2.000000
max             7.900000          4.400000  ...          2.500000
2.000000

[8 rows x 5 columns]
```

A_2.py\n====================

```
 Dataset Head
   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

 Filter Method: Variance Threshold
Selected features based on variance: ['sepal length (cm)', 'petal length
```

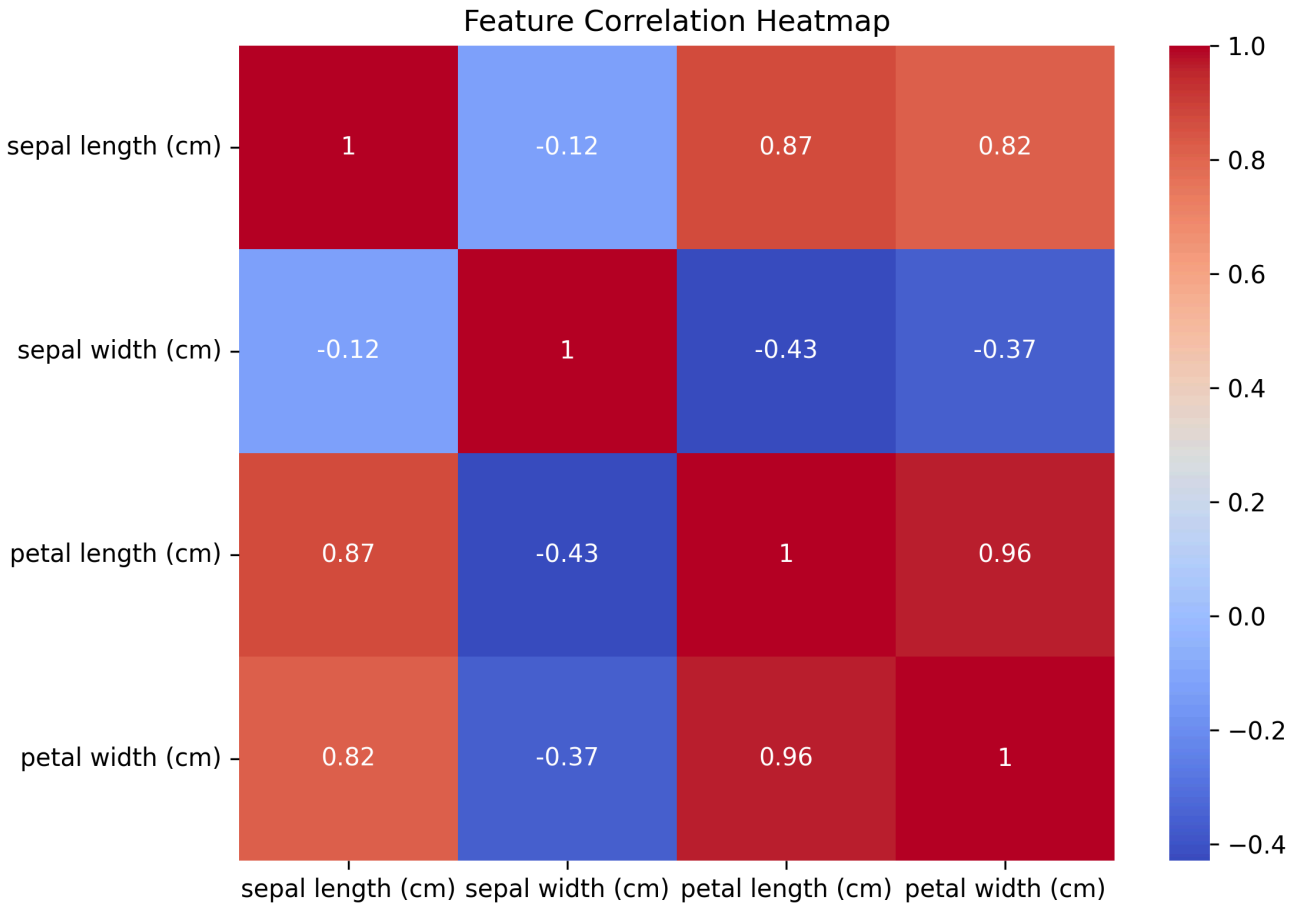
```
(cm)']

 Filter Method: Correlation
Correlation Matrix:
                   sepal length (cm)  ...  petal width (cm)
sepal length (cm)           1.000000  ...          0.817941
sepal width (cm)           -0.117570  ...         -0.366126
petal length (cm)           0.871754  ...          0.962865
petal width (cm)            0.817941  ...          1.000000

[4 rows x 4 columns]
Features after removing highly correlated: ['sepal length (cm)', 'sepal
width (cm)', 'petal length (cm)']

 Wrapper Method: RFE
Selected features by RFE: ['petal length (cm)', 'petal width (cm)']

 Embedded Method: Lasso
Selected features by Lasso: ['petal length (cm)']
```



A_3.py\n==========

```
 Original Dataset
   MedInc  HouseAge  AveRooms  ...  Latitude  Longitude  MedHouseVal
0  8.3252      41.0  6.984127  ...     37.88    -122.23        4.526
```

```
1  8.3014      21.0  6.238137  ...     37.86    -122.22        3.585
2  7.2574      52.0  8.288136  ...     37.85    -122.24        3.521
3  5.6431      52.0  5.817352  ...     37.85    -122.25        3.413
4  3.8462      52.0  6.281853  ...     37.85    -122.25        3.422

[5 rows x 9 columns]

 Dataset with Missing Values
   MedInc  HouseAge  AveRooms  ...  Latitude  Longitude  MedHouseVal
0  8.3252      41.0       NaN  ...     37.88    -122.23        4.526
1  8.3014      21.0  6.238137  ...     37.86    -122.22        3.585
2  7.2574      52.0  8.288136  ...     37.85    -122.24        3.521
3  5.6431      52.0  5.817352  ...     37.85    -122.25        3.413
4  3.8462      52.0  6.281853  ...     37.85    -122.25        3.422
5  4.0368       NaN  4.761658  ...     37.85    -122.25        2.697

[6 rows x 9 columns]

 After Handling Missing Values
   MedInc   HouseAge  AveRooms  ...  Latitude  Longitude  MedHouseVal
0  8.3252  41.000000  5.428924  ...     37.88    -122.23        4.526
1  8.3014  21.000000  6.238137  ...     37.86    -122.22        3.585
2  7.2574  52.000000  8.288136  ...     37.85    -122.24        3.521
3  5.6431  52.000000  5.817352  ...     37.85    -122.25        3.413
4  3.8462  52.000000  6.281853  ...     37.85    -122.25        3.422
5  4.0368  28.638355  4.761658  ...     37.85    -122.25        2.697

[6 rows x 9 columns]

 Standardized Features (first 5 rows)
[[ 2.34  0.98  0.   -0.15 -0.97 -0.05  1.05 -1.33]
 [ 2.33 -0.61  0.33 -0.26  0.86 -0.09  1.04 -1.32]
 [ 1.78  1.86  1.16 -0.05 -0.82 -0.03  1.04 -1.33]
 [ 0.93  1.86  0.16 -0.05 -0.77 -0.05  1.04 -1.34]
 [-0.01  1.86  0.34 -0.03 -0.76 -0.09  1.04 -1.34]]

 Min-Max Scaled Features (first 5 rows)
[[0.54 0.78 0.03 0.02 0.01 0.   0.57 0.21]
 [0.54 0.39 0.04 0.02 0.07 0.   0.57 0.21]
 [0.47 1.   0.05 0.02 0.01 0.   0.56 0.21]
 [0.35 1.   0.04 0.02 0.02 0.   0.56 0.21]
 [0.23 1.   0.04 0.02 0.02 0.   0.56 0.21]]

 Train and Test Split
X_train shape: (14448, 8)
X_test shape: (6192, 8)
y_train shape: (14448,)
y_test shape: (6192,)
```

A_4.py\n==========

```
 Dataset Head
   mean radius  mean texture  ...  worst fractal dimension  target
0        17.99         10.38  ...                  0.11890       0
1        20.57         17.77  ...                  0.08902       0
2        19.69         21.25  ...                  0.08758       0
3        11.42         20.38  ...                  0.17300       0
4        20.29         14.34  ...                  0.07678       0

[5 rows x 31 columns]

 Train-Test Split
X_train shape: (398, 30)
X_test shape: (171, 30)

 Evaluation Metrics
Accuracy: 0.9825
Precision: 0.9907
Recall: 0.9815

Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.98      0.98        63
           1       0.99      0.98      0.99       108

    accuracy                           0.98       171
   macro avg       0.98      0.98      0.98       171
weighted avg       0.98      0.98      0.98       171
```

A_5.py\n==========

```
 Dataset Head
   mean radius  mean texture  ...  worst symmetry  worst fractal dimension
0        17.99         10.38  ...          0.4601                  0.11890
1        20.57         17.77  ...          0.2750                  0.08902
2        19.69         21.25  ...          0.3613                  0.08758
3        11.42         20.38  ...          0.6638                  0.17300
4        20.29         14.34  ...          0.2364                  0.07678

[5 rows x 30 columns]

 SVM Evaluation with Different Kernels
Kernel: linear
  Accuracy: 0.9766
```

```
  Precision: 0.9815
  Recall: 0.9815

Kernel: poly
  Accuracy: 0.8947
  Precision: 0.8571
  Recall: 1.0000

Kernel: rbf
  Accuracy: 0.9766
  Precision: 0.9815
  Recall: 0.9815
```

A_6.py\n====================

```
 Comparison of ML Techniques
Logistic Regression:
  Accuracy: 0.9825
  Precision: 0.9907
  Recall: 0.9815

Decision Tree:
  Accuracy: 0.9415
  Precision: 0.9712
  Recall: 0.9352

Random Forest:
  Accuracy: 0.9708
  Precision: 0.9640
  Recall: 0.9907

SVM:
  Accuracy: 0.9766
  Precision: 0.9815
  Recall: 0.9815

KNN:
  Accuracy: 0.9591
  Precision: 0.9633
  Recall: 0.9722
```

A_7.py\n====================

```
 Dataset Head
   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

 Clustering Results
Cluster Labels for first 10 samples: [1 2 2 2 1 1 1 1 2 2]
Centroids of clusters:
[[ 0.57100359 -0.37176778  0.69111943  0.66315198]
 [-0.81623084  1.31895771 -1.28683379 -1.2197118 ]
 [-1.32765367 -0.373138   -1.13723572 -1.11486192]]
Silhouette Score: 0.4799

Original vs Cluster (first 10 rows):
   Original  Cluster
0         0        1
1         0        2
2         0        2
3         0        2
4         0        1
5         0        1
6         0        1
7         0        1
8         0        2
9         0        2
```

A_8.py\n==========

```
 Clustering Results
       Algorithm  Silhouette Score  Adjusted Rand Index
0         KMeans          0.552819             0.730238
1  Agglomerative          0.554324             0.731199

Agglomerative Clustering forms better separated clusters.
Agglomerative clustering is closer to the true Iris classes.
```

B_1.py\n==========

```
Decision Tree Rules (ID3 style):

|--- petal length (cm) <= 2.45
|   |--- class: 0
|--- petal length (cm) >  2.45
|   |--- petal length (cm) <= 4.75
|   |   |--- petal width (cm) <= 1.60
|   |   |   |--- class: 1
|   |   |--- petal width (cm) >  1.60
|   |   |   |--- class: 2
|   |--- petal length (cm) >  4.75
|   |   |--- petal length (cm) <= 5.15
|   |   |   |--- class: 2
|   |   |--- petal length (cm) >  5.15
|   |   |   |--- class: 2


Model Accuracy on Test Set: 0.9777777777777777

New Sample: [[5.1, 3.5, 1.4, 0.2]]
Predicted Class: setosa
```

B_2.py\n====================

```
Epoch 0, Loss: 0.2616
Epoch 200, Loss: 0.0164
Epoch 400, Loss: 0.0117
Epoch 600, Loss: 0.0093
Epoch 800, Loss: 0.0073

Final Accuracy on Test Data: 0.9824561403508771
```

B_3.py\n====================

```
Dataset saved to wine_dataset.csv

First 5 rows of dataset:
   alcohol  malic_acid   ash  ...  od280/od315_of_diluted_wines  proline
target
0    14.23        1.71  2.43  ...                          3.92   1065.0
0
1    13.20        1.78  2.14  ...                          3.40   1050.0
0
2    13.16        2.36  2.67  ...                          3.17   1185.0
0
3    14.37        1.95  2.50  ...                          3.45   1480.0
```

```
0
4    13.24        2.59  2.87  ...                          2.93    735.0
0

[5 rows x 14 columns]

Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

     class_0       1.00      1.00      1.00        19
     class_1       1.00      1.00      1.00        21
     class_2       1.00      1.00      1.00        14

    accuracy                           1.00        54
   macro avg       1.00      1.00      1.00        54
weighted avg       1.00      1.00      1.00        54
```

B_4.py\n====================

```
Training finished!

Average reward over 2000 episodes: 0.36

Testing trained agent:
Test finished with reward: 0.0
```

B_5.py\n=========

```
Epoch 1/5
938/938 ━━━━━━━━━━━━━━━━━━━━ 24s 24ms/step - accuracy: 0.9443 - loss:
0.1830 - val_accuracy: 0.9824 - val_loss: 0.0542
Epoch 2/5
938/938 ━━━━━━━━━━━━━━━━━━━━ 23s 24ms/step - accuracy: 0.9844 - loss:
0.0504 - val_accuracy: 0.9896 - val_loss: 0.0342
Epoch 3/5
938/938 ━━━━━━━━━━━━━━━━━━━━ 22s 24ms/step - accuracy: 0.9887 - loss:
0.0352 - val_accuracy: 0.9866 - val_loss: 0.0393
Epoch 4/5
938/938 ━━━━━━━━━━━━━━━━━━━━ 22s 24ms/step - accuracy: 0.9918 - loss:
0.0263 - val_accuracy: 0.9907 - val_loss: 0.0289
Epoch 5/5
938/938 ━━━━━━━━━━━━━━━━━━━━ 22s 24ms/step - accuracy: 0.9932 - loss:
0.0223 - val_accuracy: 0.9892 - val_loss: 0.0342
313/313 - 2s - 5ms/step - accuracy: 0.9892 - loss: 0.0342
```

```
Test Accuracy: 0.9891999959945679
```