Sri Sivasubramaniya Nadar College of Engineering, Chennai
(An autonomous Institution affiliated to Anna University)

Degree & Branch	M. Tech CSE [5 Years Integrated]	Semester	V
Subject Code & Name	ICS1512 & Machine Learning Algorithms Laboratory		
Academic year	2025-2026 (Odd)	Batch:2023-2028	**Due date:** 01-08-2025

Experiment 1: Working with Python Packages

Aim:

To explore the core functionalities of Python libraries such as NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib, and apply them to real-world datasets from public repositories like UCI and Kaggle by identifying the appropriate machine learning tasks. This includes performing essential steps in a machine learning workflow such as data loading, exploratory data analysis, preprocessing, feature selection, model building, and performance evaluation for tasks such as classification, regression, and recognition.

Library Exploration:

1. **Explore key functis and operations in Python libraries — NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib — focusing on array manipulation, data processing, machine learning, and visualization.**

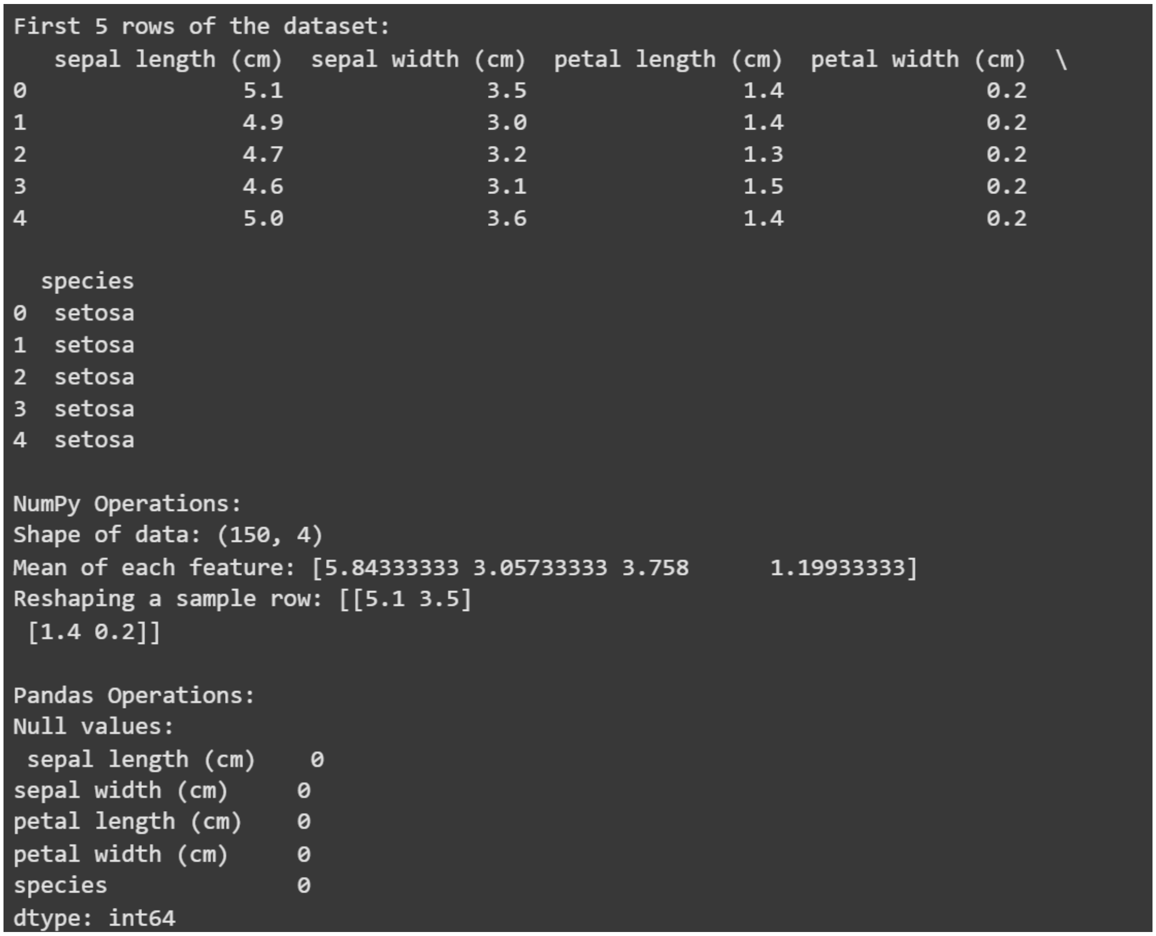
CODE:

```
# Importing required libraries
import numpy as np
import pandas as pd
from scipy import stats
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt

# 1. Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
feature_names = iris.feature_names
target_names = iris.target_names
```

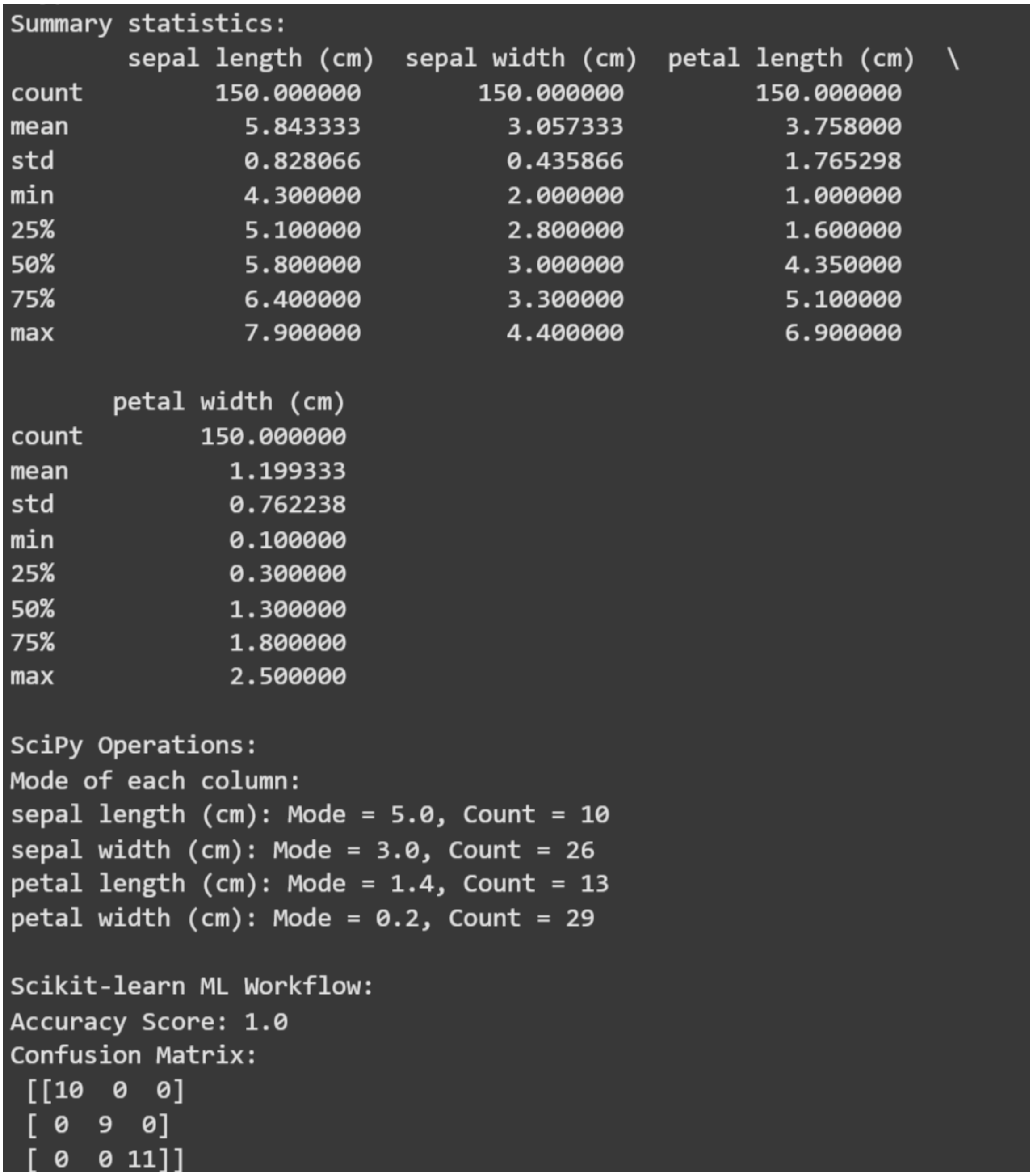
```
# Convert to DataFrame for easier handling
df = pd.DataFrame(X, columns=feature_names)
df['species'] = pd.Series([target_names[i] for i in y])

print("First 5 rows of the dataset:")
print(df.head())

# 2. NumPy: Array manipulations
print("\nNumPy Operations:")
print("Shape of data:", X.shape)
print("Mean of each feature:", np.mean(X, axis=0))
print("Reshaping a sample row:", X[0].reshape(2, 2))

# 3. Pandas: Data preprocessing
print("\nPandas Operations:")
print("Null values:\n", df.isnull().sum())
print("Summary statistics:\n", df.describe())

# Encoding species using label encoding (for general purpose)
df['species_encoded'] = pd.factorize(df['species'])[0]

# 4. SciPy: Statistical operations
print("\nSciPy Operations:")
print("Mode of each column:")
for col in feature_names:
    mode_val = stats.mode(df[col], keepdims=True)
    print(f"{col}: Mode = {mode_val.mode[0]}, Count = {mode_val.count[0]}")
    # 5. Scikit-learn: ML workflow
# Standardizing features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state

# Train a Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Predictions and evaluation
y_pred = clf.predict(X_test)
print("\nScikit-learn ML Workflow:")
print("Accuracy Score:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
# 6. Matplotlib: Data visualization
plt.figure(figsize=(10, 6))

# Scatter plot for Sepal Length vs Sepal Width
for i, label in enumerate(np.unique(y)):
    plt.scatter(X[y == i, 0], X[y == i, 1], label=target_names[i])

plt.title('Sepal Length vs Sepal Width')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.legend()
plt.grid(True)
plt.show()
```

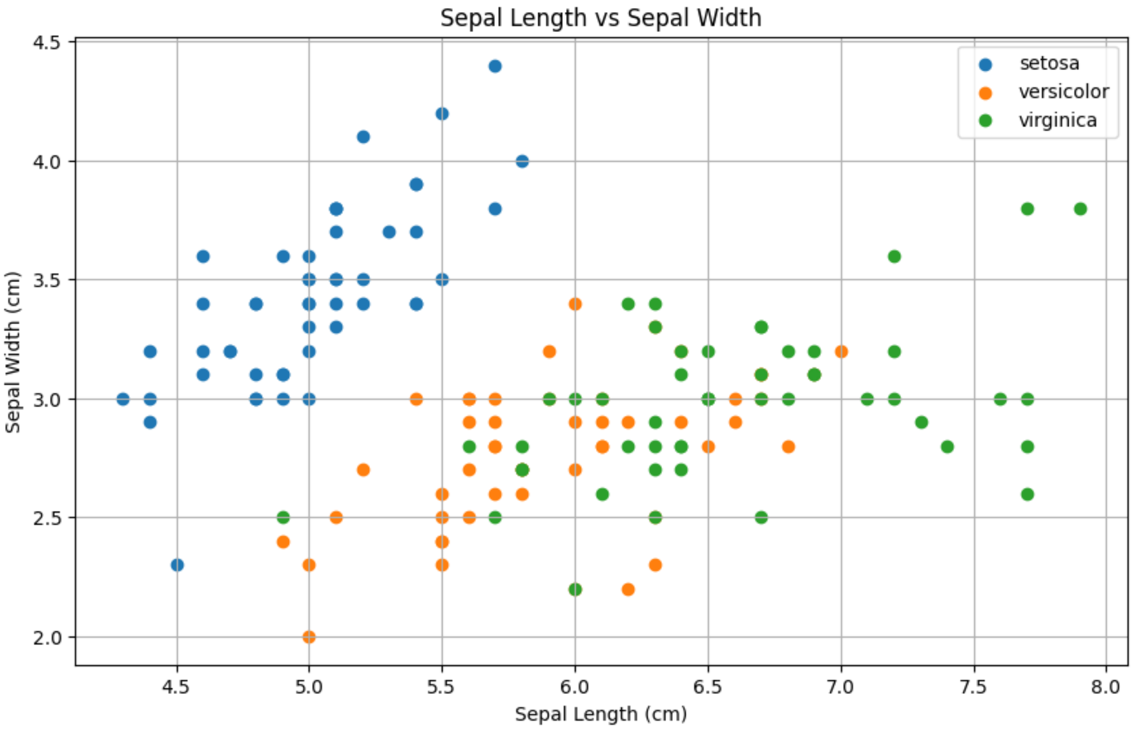
OUTPUT:



```
First 5 rows of the dataset:
   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   

  species  
0  setosa  
1  setosa  
2  setosa  
3  setosa  
4  setosa  

NumPy Operations:
Shape of data: (150, 4)
Mean of each feature: [5.84333333 3.05733333 3.758      1.19933333]
Reshaping a sample row: [[5.1 3.5]
 [1.4 0.2]]

Pandas Operations:
Null values:
 sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
species              0
dtype: int64
```



```
Summary statistics:
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)
count        150.000000
mean           1.199333
std            0.762238
min            0.100000
25%            0.300000
50%            1.300000
75%            1.800000
max            2.500000

SciPy Operations:
Mode of each column:
sepal length (cm): Mode = 5.0, Count = 10
sepal width (cm): Mode = 3.0, Count = 26
petal length (cm): Mode = 1.4, Count = 13
petal width (cm): Mode = 0.2, Count = 29

Scikit-learn ML Workflow:
Accuracy Score: 1.0
Confusion Matrix:
 [[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```


Sepal Length vs Sepal Width
● setosa
● versicolor
● virginica
Sepal Width (cm)
Sepal Length (cm)
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0

Dataset Identification:

2. **Download datasets from the UCI/Kaggle repositories and identify the type of machine learning task (e.g., classification, regression) suitable for each.**

Dataset	**Type of ML Task**	**Feature Selection Technique**	**Suitable ML Algorithm**
Iris Dataset	Classification	ANOVA, Mutual Information	Decision Tree, KNN, SVM
Loan Amount Prediction	Regression	Correlation Analysis, Lasso Regularization	Linear Regression, Random Forest Regressor
Predicting Diabetes	Classification	Recursive Feature Elimination (RFE), PCA	Logistic Regression, Random Forest
Classification of Email Spam	Classification	Chi-Square, TF-IDF, Information Gain	Naive Bayes, SVM
Handwritten Character Recognition / MNIST	Classification	PCA, Pixel Intensity Filtering	Convolutional Neural Network (CNN)

Table 1: ML Tasks, Feature Selection Techniques, and Algorithms for Different Datasets

ML Workflow Execution:

3. **Apply the machine learning workflow: load data, perform EDA, preprocess data, select features, split data, build models, and evaluate performance.**

i. **Loading the Dataset**

- Goal: Import data into the working environment.
- Tools: `pandas`, `numpy`, or database connectors.
- Example Code:

```
import pandas as pd
data = pd.read_csv("dataset.csv")
```

ii. **Exploratory Data Analysis (EDA) and Visualization**

- Goal: Understand data structure, distribution, and relationships.
- Tools: Histograms, scatter plots, bar charts, heatmaps, box plots.
- Libraries: `matplotlib`, `seaborn`, `pandas`
- Example Code:

```
import seaborn as sns
sns.heatmap(data.corr(), annot=True)
```

iii. **Data Preprocessing**

- Handle missing values (imputation or removal).
- Drop irrelevant or redundant features.
- Encode categorical variables (Label Encoding, One-Hot Encoding).
- Normalize or standardize data (e.g., using StandardScaler).

iv. **Feature Selection**

- Goal: Select the most relevant features for modeling.
- Techniques: SelectKBest, Chi-square test, ANOVA.
- Benefits: Improves performance and reduces overfitting.
- Example Code:

```
from sklearn.feature_selection import SelectKBest, chi2
X_new = SelectKBest(score_func=chi2, k=5).fit_transform(X, y)
```

v. **Data Splitting**

- Goal: Divide data into training, testing, and optionally validation sets.
- Common split ratios: 70:30 or 80:20 (Train:Test).
- Example Code:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

vi. **Performance Evaluation**

- Goal: Assess model effectiveness.
- Metrics:
  - Supervised: Accuracy, Precision, Recall, F1-score, ROC-AUC.
  - Unsupervised: Silhouette Score, Davies-Bouldin Index.
- Visual Tools: Confusion matrix, ROC curve, Precision-Recall curve.
- Libraries: `sklearn.metrics`, `matplotlib`, `seaborn`

Learning Outcomes:

- Learnt to use essential Python libraries such as NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib for performing core machine learning operations like data manipulation, computation, and visualization.
- Understood array operations and data structures in NumPy and Pandas, enabling efficient handling of large datasets.
- Explored mathematical computing functions in SciPy and applied them for statistical and scientific tasks.
- Gained hands-on experience with machine learning workflows using Scikit-learn, including preprocessing, training, and evaluating models.
- Learnt to visualize data effectively using Matplotlib through plots like histograms, bar charts, scatter plots, and heatmaps to draw meaningful insights.
- Explored and analyzed real-world datasets from public repositories like UCI and Kaggle and identified appropriate ML models and tasks (classification, regression, etc.).
- Developed an understanding of the end-to-end ML pipeline, including loading data, EDA, preprocessing, feature selection, model training, and evaluation using suitable metrics.