Experiment 1: Working with Python Packages for Machine Learning

Ramlath Nisha A
Email: ramlathnisha2310611@ssn.edu.in

***Abstract*—This experiment explores essential Python libraries used in machine learning workflows, namely NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib. The objective is to understand key operations such as array manipulation, data preprocessing, mathematical computing, machine learning model development, and data visualization. Multiple benchmark datasets—including Iris, loan amount prediction, email spam/MNIST, and diabetes prediction—are examined to identify the corresponding type of machine learning task, suitable feature selection techniques, and appropriate algorithms. For selected datasets (e.g., Iris and loan amount prediction), complete end-to-end workflows are implemented in Python, covering data loading, exploratory data analysis (EDA), preprocessing, feature selection, train–test splitting, model training, evaluation, and visualization. The findings are summarized in task and inference tables, followed by a reflection on learning outcomes.**

***Index Terms*—NumPy, Pandas, SciPy, Scikit-learn, Matplotlib, Machine Learning Workflow, Feature Selection, Supervised Learning, Regression, Classification**

I. Introduction

Python has become a dominant language for data science and machine learning due to its rich ecosystem of open-source libraries. In particular:

- **NumPy** enables efficient numerical computations using multi-dimensional arrays.
- **Pandas** provides powerful data structures and tools for data cleaning and manipulation.
- **SciPy** extends numerical routines to scientific computing and statistics.
- **Scikit-learn** standardizes machine learning workflows and algorithms.
- **Matplotlib** facilitates flexible data visualization.

In this experiment, these libraries are used to construct simple machine learning workflows on several benchmark datasets. For each dataset, the type of machine learning task (e.g., supervised classification, regression) is identified, and the typical workflow steps—data loading, EDA, preprocessing, feature selection, splitting, and evaluation—are explored. For selected datasets, such as the Iris data and the loan amount prediction data, a complete implementation is carried out in the notebook `EX 1 Full.ipynb`, including model training and performance analysis.

II. Aim and Objectives

A. Aim

To explore core Python libraries for machine learning and to apply them to multiple benchmark datasets, identifying the appropriate type of machine learning task, feature selection techniques, and suitable algorithms.

B. Objectives

- To study the basic functions and methods in NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib.
- To explore public datasets from UCI and Kaggle and classify them into appropriate machine learning tasks (supervised/unsupervised, regression/classification).
- To understand and implement the main steps in a machine learning workflow: loading data, EDA, preprocessing, feature selection, data splitting, and evaluation.
- To implement complete workflows for selected datasets (e.g., Iris classification and loan amount regression) in Python.
- To summarize observations in task and inference tables and reflect on learning outcomes.

III. Software and Libraries Used

A. Software

- Python 3.x
- Jupyter Notebook (`EX 1 Full.ipynb`)
- Anaconda/Miniconda distribution (for package management)

B. Python Libraries

- **NumPy**: Numerical array operations, broadcasting, and linear algebra.
- **Pandas**: DataFrame and Series objects for tabular data, data cleaning, and manipulation.
- **SciPy**: Scientific computing and statistical functions.
- **Scikit-learn**: Preprocessing, feature selection, model training, cross-validation, and evaluation.
- **Matplotlib** (+ Seaborn): Data visualization and plotting.

IV. Dataset Description and ML Task Identification

A. Datasets Considered

The following datasets are explored conceptually and/or implemented:

- **Iris Dataset**: Flower measurements for three species of iris.
- **Loan Amount Prediction**: Loan application features with loan amount as target.
- **Email Spam / MNIST Data**: Numerical features extracted from emails or image pixels for digit recognition.

- **Predicting Diabetes**: Medical data to classify individuals as diabetic or non-diabetic.

B. Summary of Task Type, Feature Selection, and Algorithm

Table **??** summarizes the type of machine learning task associated with each dataset, feature selection techniques, and suitable algorithms.

TABLE I
SUMMARY OF ML TASK, FEATURE SELECTION, AND SUITABLE ALGORITHM

Dataset	Type of ML Task	Feature Selection	Suitable ML Algorithm
Iris Dataset	Classification	SelectKBest	Logistic Reg.
Loan Amount	Regression	SelectKBest	Linear Reg.
Diabetes	Classification	SelectKBest	Logistic Reg.
Email Spam	Classification	Chi-square	Logistic Reg.
Handwritten Digits	Classification	PCA	KNN

V. MACHINE LEARNING WORKFLOW

The general workflow followed for each dataset is as follows.

A. Loading the Dataset

- Datasets are loaded using `pandas.read_csv()` (e.g., Iris data from `iris.data`, loan dataset from CSV files).
- Column names are assigned explicitly for the Iris dataset:
  - SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm, Species.

B. Exploratory Data Analysis and Visualization

- Summary statistics (shape, missing values, basic statistics) are inspected using `data.shape`, `data.isna().sum()`, and `data.describe()`.
- Visual tools include histograms, scatter plots, box plots, pair plots, and correlation heatmaps.

C. Data Preprocessing

- Missing values (if present) are handled via imputation or removal.
- Outliers in numerical features are detected using the IQR rule and removed:

$$\text{IQR} = Q3 - Q1,$$
$$\text{Range} = [Q1 - 1.5 \cdot \text{IQR},\ Q3 + 1.5 \cdot \text{IQR}].$$

- Categorical variables (e.g., Species in the Iris dataset) are encoded using `LabelEncoder`.
- For some datasets, numerical features are standardized/normalized prior to model training.

D. Feature Selection

- For tabular datasets, filter-based methods such as SelectKBest with ANOVA F-test or chi-square test can be applied.
- For high-dimensional data (e.g., MNIST), dimensionality reduction via PCA is appropriate.

E. Data Splitting

- Data is split into training and testing sets using `train_test_split()`.
- For the Iris dataset, a typical split is 80% training and 20% testing with stratified sampling.

F. Performance Evaluation

- Classification metrics: accuracy, precision, recall, F1-score, confusion matrix, and class-wise reports.
- Regression metrics (for loan amount prediction): MAE, MSE, RMSE, and R^2.
- Cross-validation is used (e.g., 5-fold KFold) to estimate model performance more robustly.

VI. IMPLEMENTATION DETAILS (EX 1 FULL NOTEBOOK)

A. Iris Dataset: Multi-class Classification

In `EX 1 Full.ipynb`, the Iris dataset is processed as follows:

- **Loading**: Data is read from `iris.data` into a Pandas DataFrame with named columns.
- **EDA**: Dataset shape and missing values are printed; distributions of numerical features (sepal and petal lengths/widths) are plotted using histograms with KDE.
- **Outlier Removal**: The IQR rule is applied to remove outliers from each numerical column.
- **Encoding**: Species labels are encoded to integers using `LabelEncoder`.
- **Feature and Target**: X is defined as all numerical features; y is the encoded Species.
- **Train–Test Split**: Data is split into training and test sets with stratification.
- **Models**: Logistic Regression and Random Forest classifiers are defined.
- **Cross-Validation**: 5-fold cross-validation (KFold) is used to estimate accuracy of both models.
- **Training and Testing**: Logistic Regression (best model) is trained on the training set and evaluated on the test set.
- **Evaluation**: Test accuracy and a detailed classification report are printed. A confusion matrix is plotted using Seaborn heatmap.
- **Sample Prediction**: A sample input is created, and the model predicts the Iris species, which is then decoded back to the original label.

B. Loan Amount Prediction: Regression

For the loan amount prediction dataset (as indicated in the notebook):

- **Loading**: The training data is loaded and columns related to applicant information, credit history, and other relevant features are read.
- **EDA**: Summary statistics and basic plots (e.g., histograms and box plots) are used to understand the distribution of features and target (loan amount).
- **Preprocessing**: Missing values are handled, categorical features are encoded, and numerical features may be scaled.

- **Feature Selection**: Relevant features are selected based on domain knowledge and correlation analysis.
- **Model**: A regression model such as Gradient Boosting Regressor (GBR) is trained to predict loan amount.
- **Evaluation**: Performance is measured using regression metrics and visualized using scatter plots of predicted vs. actual loan amounts.

C. Other Datasets (Conceptual Workflow)

For the remaining datasets (email spam/MNIST and diabetes prediction), a similar workflow is conceptually applied:

- Identify task type (binary/multi-class classification).
- Perform EDA and preprocessing (handling missing values, encoding categoricals, scaling).
- Apply appropriate feature selection techniques (e.g., chi-square, SelectKBest, PCA).
- Train baseline models such as Logistic Regression, Naïve Bayes, Random Forest, KNN, or SVM.
- Evaluate using classification metrics and confusion matrices.

VII. Visualizations

This section provides placeholders for key visualizations generated using Matplotlib/Seaborn. Replace filenames with your saved figures.

A. Iris Dataset Visualizations

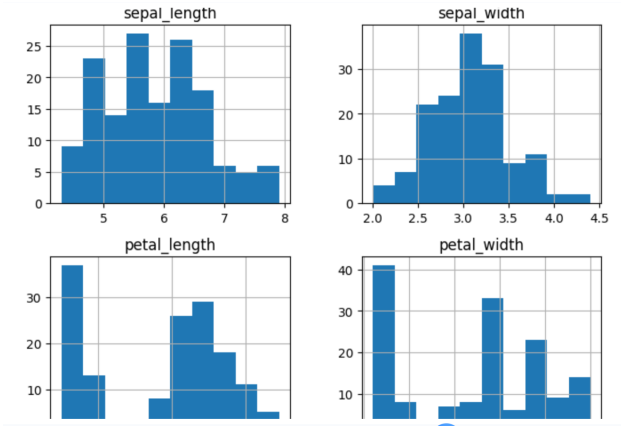

Fig. 1. Histograms of Iris feature distributions (sepal and petal lengths/widths).

B. Loan Amount Prediction Visualizations

C. Additional Visualizations

VIII. Inference and Summary Tables

A. Inference Table

Table II summarizes key inferences from each dataset.

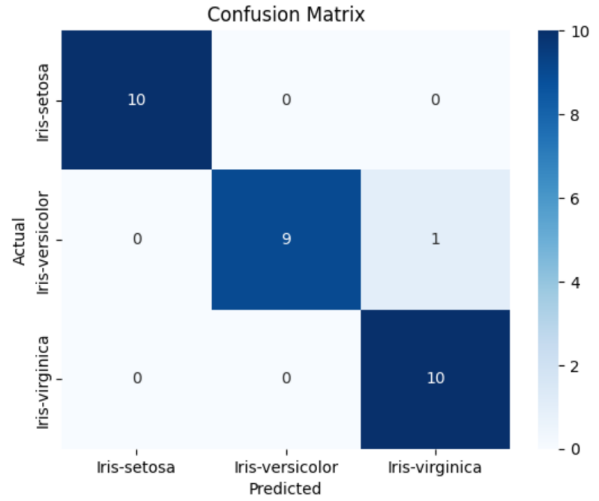

Fig. 2. Confusion matrix for Iris classification (Logistic Regression).

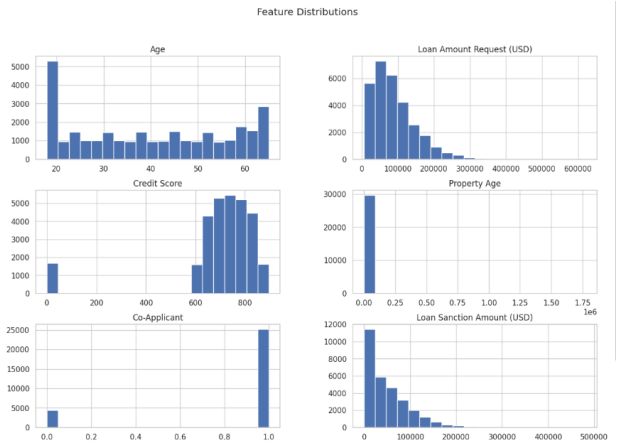

Fig. 3. Example feature distributions for the loan amount dataset.

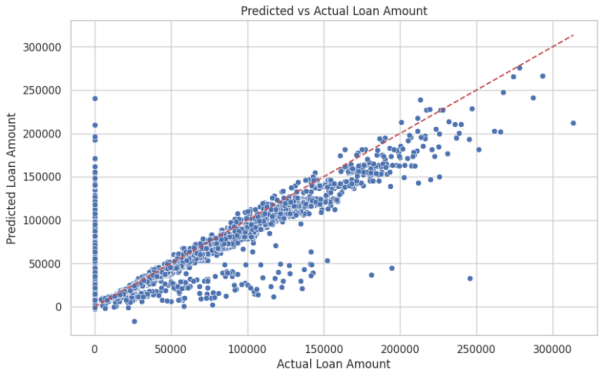

Fig. 4. Predicted vs. actual loan amounts for regression model.

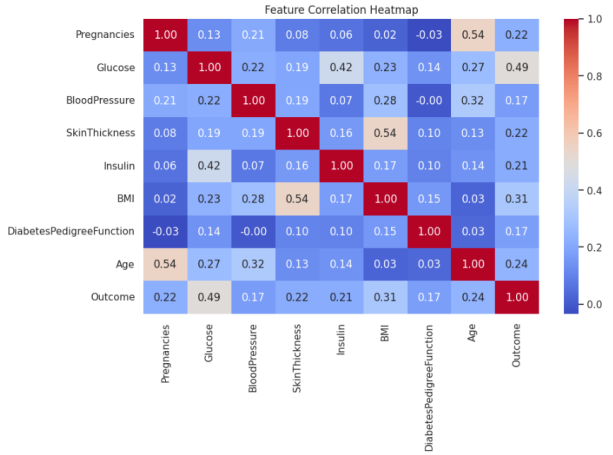

Fig. 5. Correlation heatmap for diabetes prediction dataset.

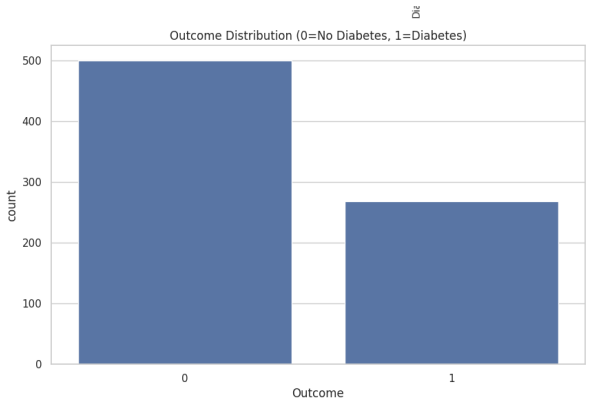

Fig. 6. Sample handwritten digit images (MNIST).

TABLE II
INFERENCE SUMMARY FOR EACH DATASET

Dataset	Key Inference
Iris Dataset	Classes are well-separated in feature space; simple models like Logistic Regression and Random Forest achieve high accuracy with minimal preprocessing and outlier removal.
Loan Amount Prediction	Loan amount is influenced by multiple applicant and loan-related features; regression models (e.g., Gradient Boosting) can capture non-linear relationships, and preprocessing strongly affects performance.
Predicting Diabetes	Class imbalance and overlapping features require careful threshold selection and evaluation using precision, recall, and F1-score rather than accuracy alone.
Email Spam Classification	Text-derived numerical features distinguish spam from ham; Naïve Bayes and linear classifiers generally perform well after appropriate preprocessing and feature selection.

IX. Reflection on Learning Outcomes

A. Key Learning Points

- Developed familiarity with core Python packages (NumPy, Pandas, SciPy, Scikit-learn, Matplotlib) and their roles in a machine learning pipeline.
- Understood how to load, explore, and preprocess different kinds of datasets.
- Learned to identify appropriate machine learning task types and algorithms based on dataset characteristics.
- Practiced feature selection and outlier handling to improve model performance.
- Used cross-validation and performance metrics to evaluate and compare models.

B. Reflection

- The experiment emphasized that data understanding and preprocessing are as important as selecting algorithms.
- Working with multiple datasets provided exposure to both regression and classification tasks, enhancing generalization of the workflow.
- Implementing full pipelines in Jupyter notebooks improved coding skills and confidence in using Python for machine learning.
- The experiment laid a strong foundation for more advanced tasks such as regularized regression, Naïve Bayes, KNN, and deep learning in later experiments.

X. Conclusion

In this experiment, we explored fundamental Python libraries that form the backbone of modern machine learning workflows. Using NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib, we implemented and analyzed complete pipelines for datasets such as Iris and loan amount prediction and conceptually extended the workflow to others like diabetes prediction, email spam, and MNIST.

We identified the type of machine learning task associated with each dataset, selected appropriate feature selection techniques, and proposed suitable algorithms. Through EDA, preprocessing, and careful evaluation, we observed how these steps directly impact model performance and interpretability. The knowledge gained here serves as an essential prerequisite for subsequent experiments focusing on specific algorithms, hyperparameter tuning, and deeper analyses of overfitting, underfitting, and bias–variance trade-offs.