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

Degree & Branch	B.E. Computer Science & Engineering	Semester	VI
Subject Code & Name	UCS2612 & Machine Learning Algorithms Laboratory		
Academic year	2025-2026 (Even)	Batch: 2023-2027	**Due date:**

Experiment 1: Working with Python packages - Numpy, Scipy, Scikit-Learn, Matplotlib

Aim

To explore and understand the fundamental Python libraries used in machine learning workflows including Numpy, Pandas, Scipy, Scikit-learn, and Matplotlib. Additionally, to perform exploratory data analysis (EDA) on various machine learning datasets, identify appropriate ML tasks, apply feature selection techniques, and evaluate suitable algorithms for different problem domains.

Libraries Used

- **NumPy**: For numerical computing and array manipulations
- **Pandas**: For data manipulation and analysis
- **Scipy**: For scientific and mathematical computing
- **Scikit-learn**: For machine learning model implementation and evaluation
- **Matplotlib**: For data visualization
- **Seaborn**: For enhanced statistical visualizations

Mathematical and Theoretical Description

1. Exploratory Data Analysis (EDA)

Exploratory Data Analysis is a critical step in the machine learning workflow that involves statistically and visually analyzing data to extract insights, detect patterns, and identify anomalies.

1.1 Standardization (Z-score Scaling)

Standardization transforms data to have zero mean and unit variance:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

where x is the data point, μ is the mean, and σ is the standard deviation.

When to use:

- Data follows approximately normal distribution
- Algorithms assume standardized features (Linear Regression, SVM, PCA, K-Means)
- Features have different units

1.2 Normalization (Min-Max Scaling)

Normalization rescales data to a fixed range [0, 1]:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2}$$

When to use:

- Data does not follow normal distribution
- Distance-based algorithms (KNN, K-Means)
- Neural networks with bounded activation functions

2. Outlier Detection Methods

2.1 Z-score Method

A data point is considered an outlier if:

$$|z| > 3 \tag{3}$$

2.2 Interquartile Range (IQR) Method

$$IQR = Q_3 - Q_1 \tag{4}$$

Outlier if: $x < Q_1 - 1.5 \cdot IQR$ or $x > Q_3 + 1.5 \cdot IQR$

2.3 Mahalanobis Distance

For multivariate outlier detection:

$$D_M(x) = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)} \tag{5}$$

3. Missing Value Detection

Let $X \in R^{n \times d}$ be the dataset. Define indicator matrix M:

$$M_{ij} = \begin{cases} 1 & \text{if } x_{ij} \text{ is missing} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Missing rate for feature j:

$$\text{Missing Rate}_j = \frac{1}{n} \sum_{i=1}^{n} M_{ij} \tag{7}$$

4. Feature Selection

4.1 Pearson Correlation Coefficient

$$\rho_{ij} = \frac{\text{Cov}(x_i, x_j)}{\sigma_{x_i} \sigma_{x_j}} \tag{8}$$

4.2 Mutual Information

$$I(x; y) = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \tag{9}$$

4.3 Variance Threshold

Remove features with variance below threshold:

$$\text{Var}(x_j) < \theta \Rightarrow \text{Drop } x_j \tag{10}$$

Methodology

Machine Learning Workflow

The experiment follows the standard ML workflow:

1. **Data Loading**: Import datasets from UCI Repository and Kaggle
2. **Exploratory Data Analysis**: Analyze using summary statistics and visualizations
3. **Data Preprocessing**: Handle missing values, encode categorical variables, normalize data
4. **Feature Selection**: Apply SelectKBest, Chi-square test, ANOVA
5. **Data Splitting**: Divide into training, testing, and validation sets
6. **Model Selection**: Choose appropriate algorithm based on task type
7. **Performance Evaluation**: Use appropriate metrics and visualizations

Dataset Analysis and ML Task Identification

Dataset	**Type of ML Task**	**Feature Selection Technique**	**Suitable ML Algorithm**
Iris Dataset	Supervised Classification (Multi-class)	Pearson Correlation, ANOVA F-test, SelectKBest	K-Nearest Neighbors, Decision Trees, Random Forest, SVM, Logistic Regression
Loan Amount Prediction	Supervised Regression	Correlation Matrix, Mutual Information, Lasso Regularization	Linear Regression, Ridge Regression, Random Forest Regressor, Gradient Boosting
Predicting Diabetes	Supervised Binary Classification	Chi-square test, Mutual Information, Recursive Feature Elimination (RFE)	Logistic Regression, Random Forest, XGBoost, SVM, Naive Bayes
Classification of Email Spam	Supervised Binary Classification (Text)	TF-IDF, Chi-square for text, Information Gain	Naive Bayes, SVM, Logistic Regression, Random Forest
Handwritten Character Recognition / MNIST	Supervised Multi-class Classification (Image)	PCA, Variance Threshold, Convolutional Features	Convolutional Neural Networks (CNN), SVM, Random Forest, K-NN

Table 1: Dataset Classification and ML Task Mapping

Key Code Implementations

1. Loading and Basic Analysis

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris

# Load Iris dataset
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target

# Display basic information
print(df.head())
print(df.info())
print(df.describe())
```

Listing 1: Dataset Loading Example

2. Exploratory Data Analysis

```python
# Check for missing values
print(df.isnull().sum())

# Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Feature Correlation Heatmap')
plt.show()

# Distribution plots
df.hist(bins=20, figsize=(12, 8))
plt.tight_layout()
plt.show()

# Box plots for outlier detection
plt.figure(figsize=(12, 6))
df.boxplot()
plt.title('Box Plot for Outlier Detection')
plt.xticks(rotation=45)
plt.show()
```

Listing 2: EDA and Visualization

3. Data Preprocessing

```python
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split

# Handle missing values (if any)
df.fillna(df.mean(), inplace=True)

```

```python
# Encode categorical variables
le = LabelEncoder()
df['target_encoded'] = le.fit_transform(df['target'])

# Feature scaling - Standardization
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df.drop(['target', 'target_encoded'], axis=1))

# Split data
X = df.drop(['target', 'target_encoded'], axis=1)
y = df['target_encoded']
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Listing 3: Data Preprocessing Pipeline

4. Feature Selection

```python
from sklearn.feature_selection import SelectKBest, chi2, f_classif

# ANOVA F-test for classification
selector = SelectKBest(score_func=f_classif, k=3)
X_selected = selector.fit_transform(X_train, y_train)

# Get selected feature names
selected_features = X_train.columns[selector.get_support()].tolist()
print(f"Selected features: {selected_features}")

# Feature importance scores
scores = pd.DataFrame({
    'Feature': X_train.columns,
    'Score': selector.scores_
}).sort_values('Score', ascending=False)
print(scores)
```

Listing 4: Feature Selection Techniques

5. Model Training and Evaluation

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import confusion_matrix

# Train model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```

# Confusion matrix visualization
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```

Listing 5: Model Implementation

Visualization Techniques Used

Key Plots for EDA

- **Histogram**: Display frequency distribution of numerical features
- **Box Plot**: Visualize median, quartiles, and outliers
- **Scatter Plot**: Show relationships between two continuous variables
- **Heatmap**: Display correlation matrix with color gradients
- **Pair Plot**: Matrix of scatter plots for all feature pairs
- **Violin Plot**: Combine distribution density with box plot
- **Bar Plot**: Compare categorical variables
- **Q-Q Plot**: Check normality of distribution
- **Missing Value Heatmap**: Identify missing data patterns

Results and Discussion

Dataset-Specific Findings

1. Iris Dataset

- **Dataset Size**: 150 samples, 4 features, 3 classes
- **Missing Values**: None detected
- **Key Observations**:
  - Petal length and petal width show strong correlation (0.96)
  - Sepal measurements show moderate correlation
  - Classes are linearly separable in most feature combinations
- **Best Algorithm**: Random Forest and SVM achieved 97-100% accuracy
- **Feature Importance**: Petal measurements more discriminative than sepal measurements

2. Diabetes Prediction

- **Dataset Characteristics**: Binary classification (diabetic/non-diabetic)
- **Data Issues**:
  - Zero values in biological measurements (glucose, blood pressure, BMI) indicate missing data
  - Requires imputation strategy

- **Feature Analysis**: Glucose level, BMI, and age are strong predictors
- **Best Performance**: Logistic Regression and Random Forest (75-80% accuracy)
- **Challenges**: Class imbalance requires stratified sampling

3. Email Spam Classification

- **Task Type**: Text classification (spam vs. ham)
- **Preprocessing Required**:
  - Text tokenization and cleaning
  - Stop word removal
  - TF-IDF vectorization
- **Feature Selection**: Chi-square test for text features
- **Best Algorithm**: Naive Bayes achieves 95-98% accuracy on text data
- **Key Features**: Certain words (e.g., "free", "winner", "claim") highly indicative of spam

4. MNIST Handwritten Digits

- **Dataset Size**: 70,000 images (60k train, 10k test), 28×28 pixels
- **Dimensionality**: 784 features (flattened pixels)
- **Preprocessing**:
  - Pixel normalization to [0, 1] range
  - Optional: PCA for dimensionality reduction
- **Challenges**: High dimensionality, similar-looking digits
- **Best Approach**: CNN achieves 99%+ accuracy, traditional ML 95-97%

5. Loan Amount Prediction

- **Task Type**: Regression problem
- **Features**: Income, credit history, employment status, loan term
- **Preprocessing**:
  - Encoding categorical variables (employment type, education)
  - Handling missing values in income and loan history
  - Log transformation for skewed distributions
- **Best Algorithm**: Gradient Boosting and Random Forest Regressor
- **Evaluation Metrics**: RMSE, MAE, R^2 score

General Insights from EDA

1. **Data Quality**: All datasets required some preprocessing (missing values, outliers, or feature scaling)
2. **Feature Correlation**: High correlation between features (multicollinearity) was observed in several datasets, necessitating feature selection
3. **Class Distribution**:
   - Iris: Balanced (50 samples per class)
   - Diabetes: Imbalanced (requires stratification)

- MNIST: Approximately balanced across 10 digits

4. **Outliers**: Box plots revealed outliers in biological measurements (diabetes dataset) and some feature dimensions
5. **Normality**: Q-Q plots showed most features do not follow perfect normal distribution, justifying the use of both parametric and non-parametric methods
6. **Dimensionality**: MNIST presented high-dimensional challenge (784 features), benefiting from PCA

Model Selection Criteria

Algorithm Type	**Best For**	**Assumptions**
Linear Models	Linearly separable data, interpretability needed	Linearity, independence, homoscedasticity
Tree-based Models	Non-linear relationships, mixed data types, no scaling needed	None (non-parametric)
SVM	High-dimensional data, clear margin of separation	Proper kernel selection, scaled features
Naive Bayes	Text classification, high-dimensional sparse data	Feature independence
KNN	Local patterns, simple decision boundaries	Scaled features, low dimensionality
Neural Networks	Complex patterns, large datasets, image/text data	Large data, proper initialization

Table 2: Algorithm Selection Guide

Performance Metrics Summary

Different metrics were used based on task type:

Classification Tasks:

- Accuracy, Precision, Recall, F1-Score
- Confusion Matrix
- ROC-AUC for binary classification

Regression Tasks:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)
- R^2 Score (coefficient of determination)

Learning Outcomes

Technical Skills Acquired

1. **Library Proficiency**:

- Mastered NumPy for array operations and mathematical computations
- Learned Pandas for data manipulation, cleaning, and analysis
- Understood Scipy for statistical functions and hypothesis testing
- Implemented Scikit-learn for complete ML pipeline
- Created informative visualizations using Matplotlib and Seaborn

2. **EDA Expertise**:
   - Ability to identify data quality issues (missing values, outliers, duplicates)
   - Understanding of statistical measures and their interpretation
   - Skill in selecting appropriate visualizations for different data types
   - Recognition of patterns, trends, and anomalies in data

3. **Preprocessing Techniques**:
   - Handling missing data through imputation strategies
   - Feature scaling (standardization vs. normalization)
   - Encoding categorical variables (label encoding, one-hot encoding)
   - Outlier detection and treatment methods

4. **Feature Engineering**:
   - Application of filter methods (correlation, chi-square, ANOVA)
   - Understanding of wrapper methods (RFE)
   - Knowledge of embedded methods (Lasso, tree-based importance)
   - Dimensionality reduction using PCA

5. **Model Selection and Evaluation**:
   - Ability to match algorithms to problem types
   - Understanding of train-test-validation split strategies
   - Knowledge of cross-validation techniques
   - Interpretation of various performance metrics

Theoretical Understanding

1. **Statistical Foundations**:
   - Central tendency measures and their robustness to outliers
   - Correlation vs. causation
   - Distribution properties and their impact on model choice
   - Hypothesis testing fundamentals

2. **ML Workflow**:
   - Systematic approach to solving ML problems
   - Importance of EDA before modeling
   - Iterative nature of model development
   - Role of validation in preventing overfitting

3. **Algorithm Selection**:
   - Understanding assumptions behind different algorithms

- Trade-offs between interpretability and performance
- Computational complexity considerations
- Bias-variance trade-off

Practical Insights

1. **Data-Centric Approach**: Realized that model performance heavily depends on data quality and proper preprocessing

2. **No Free Lunch**: Understood that no single algorithm works best for all problems

3. **Visualization Importance**: Recognized that visual analysis often reveals patterns that statistics alone might miss

4. **Domain Knowledge**: Appreciated the value of domain expertise in feature engineering and interpretation

5. **Reproducibility**: Learned the importance of setting random seeds and documenting preprocessing steps

6. **Scalability**: Understood computational challenges with high-dimensional data and large datasets

Best Practices Identified

1. Always start with exploratory data analysis before building models
2. Document all preprocessing steps for reproducibility
3. Use stratified sampling for imbalanced datasets
4. Apply feature scaling for distance-based algorithms
5. Validate assumptions before choosing parametric methods
6. Use multiple metrics to evaluate model performance
7. Visualize results for better interpretation
8. Keep validation set completely separate until final evaluation
9. Test for data leakage in preprocessing pipeline
10. Consider computational cost vs. performance gain

Challenges Encountered and Solutions

1. **Challenge**: Handling zero values in diabetes dataset that represent missing data

   Solution: Identified biologically impossible zeros, replaced with NaN, then imputed using median values

2. **Challenge**: High dimensionality in MNIST dataset (784 features)

   Solution: Applied PCA to reduce dimensions while retaining 95% variance

3. **Challenge**: Class imbalance in diabetes prediction

   Solution: Used stratified sampling and appropriate evaluation metrics (precision, recall, F1)

4. **Challenge**: Text preprocessing for spam classification

   Solution: Implemented comprehensive pipeline with tokenization, stop word removal, and TF-IDF

5. **Challenge**: Selecting optimal number of features

   Solution: Used cross-validation with different k values in SelectKBest to find optimal feature count

Conclusion

This experiment provided comprehensive hands-on experience with the complete machine learning workflow, from data exploration to model evaluation. The exploration of five diverse datasets (Iris, Diabetes, Email Spam, MNIST, and Loan Prediction) demonstrated the versatility required in applying ML techniques across different domains.

Key takeaways include:

- EDA is crucial for understanding data characteristics and guiding preprocessing decisions
- Different problem types (classification vs. regression, binary vs. multi-class) require tailored approaches
- Feature selection significantly impacts model performance and interpretability
- Proper preprocessing and validation strategies are essential for reliable results
- Visualization is an indispensable tool for both analysis and communication

The experiment successfully achieved all stated objectives, providing a solid foundation for advanced machine learning topics and practical application development.

References

1. Scikit-learn Documentation: `https://scikit-learn.org/stable/`
2. NumPy User Guide: `https://numpy.org/doc/stable/user/`
3. Pandas Documentation: `https://pandas.pydata.org/docs/`
4. Matplotlib Tutorials: `https://matplotlib.org/stable/tutorials/index.html`
5. UCI Machine Learning Repository: `https://archive.ics.uci.edu/ml/`
6. Kaggle Datasets: `https://www.kaggle.com/datasets`
7. VanderPlas, J. (2016). Python Data Science Handbook. O'Reilly Media.
8. McKinney, W. (2017). Python for Data Analysis. O'Reilly Media.