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

Degree & Branch:	B.E. Computer Science & Engineering
Semester:	V
Subject Code & Name:	ICS1512 & Machine Learning Algorithms Laboratory
Academic year:	2025-2026 (Odd)
Name:	Mohammed Aatif
Registration No:	**3122237001026**

1 Introduction

This report presents the results of machine learning experiments conducted using Python. The aim was to explore Python libraries such as NumPy, Pandas, SciPy, Scikit-learn, and Matplotlib, and apply them to various machine learning workflows on datasets like Iris, Loan, Diabetes, Spam, and MNIST.

Evaluation metrics such as ROC curve, Confusion Matrix, and Feature Importance were used. Visualizations were generated using Matplotlib and Seaborn.

2 Iris Dataset - Classification

2.1 Python Code

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix

iris = sns.load_dataset('iris')
sns.pairplot(iris, hue='species')
plt.show()

X = iris.drop('species', axis=1)
y = iris['species']
X_new = SelectKBest(score_func=f_classif, k='all').fit_transform(X, y)

X_train, X_test, y_train, y_test = train_test_split(
    X_new, y, test_size=0.2, random_state=42)

model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

```
22
23 print(confusion_matrix(y_test, y_pred))
24 print(classification_report(y_test, y_pred))
```

Listing 1: Logistic Regression on Iris Dataset

2.2 Results

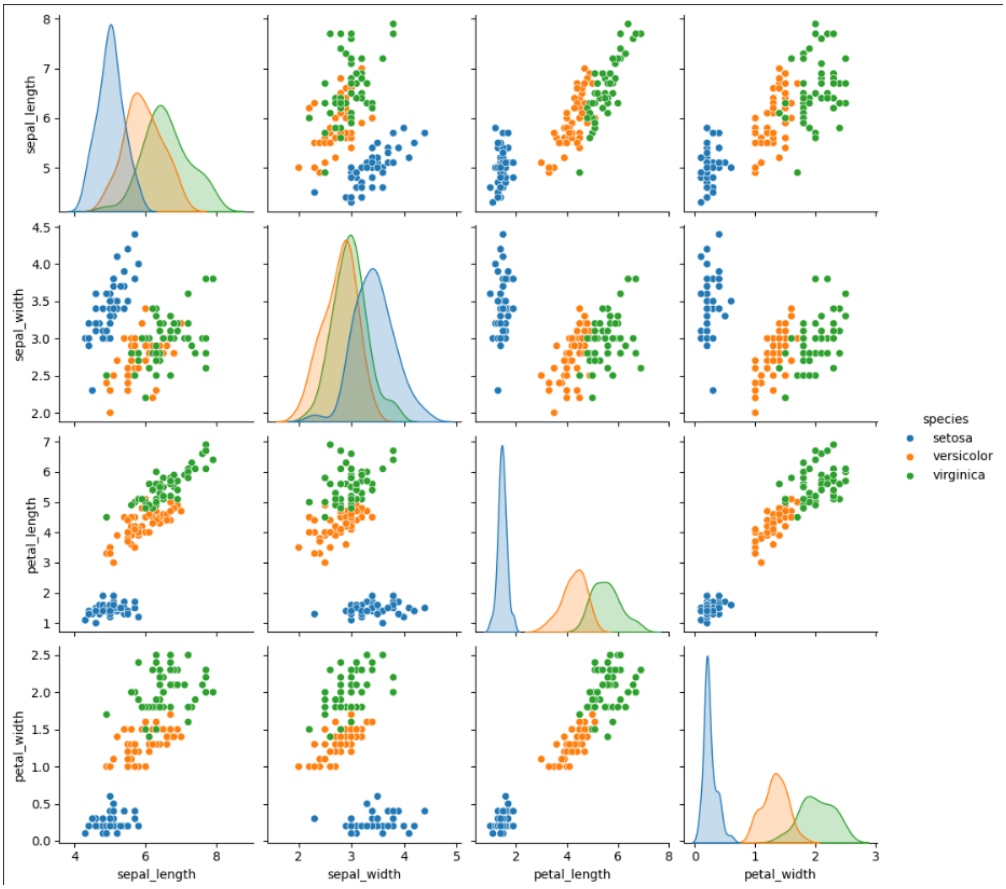

Figure 1: Pairplot of Iris dataset showing feature distributions and class separation.

- Algorithm: Logistic Regression
- Result: 97% accuracy with strong precision and recall across all classes.

3 Loan Amount Prediction - Regression

3.1 Python Code

```
1 import pandas as pd
2 from sklearn.model_selection import train_test_split
3 from sklearn.linear_model import LinearRegression
4 from sklearn.preprocessing import StandardScaler
```

```
from sklearn.metrics import mean_squared_error, r2_score

df = pd.read_csv('/content/drive/MyDrive/train.csv')
df.drop(columns=["Customer ID", "Name", "Property ID"], inplace=True)
df.dropna(inplace=True)

X = pd.get_dummies(df.drop(columns=["Loan Sanction Amount (USD)"]), drop_first=
    True)
y = df["Loan Sanction Amount (USD)"]
X = StandardScaler().fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print("RMSE:", mean_squared_error(y_test, y_pred))
print("R2 Score:", r2_score(y_test, y_pred))
```

Listing 2: Linear Regression for Loan Prediction

3.2 Results

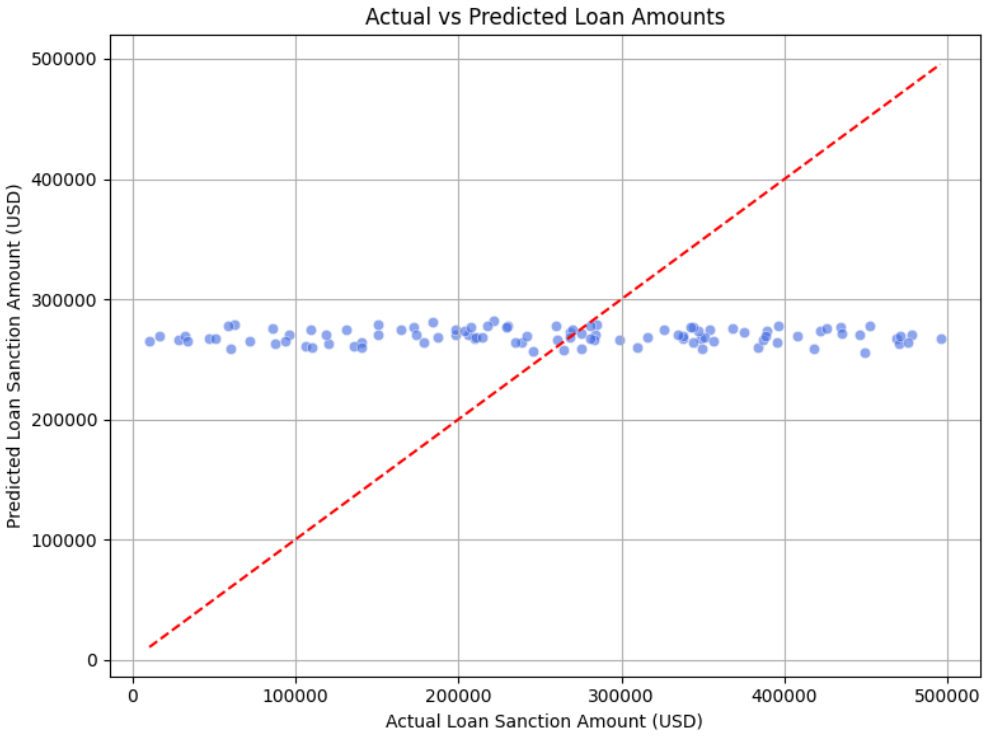

Figure 2: Actual vs Predicted Loan Sanction Amounts.

- RMSE = 1.19B, $R^2 = 0.47$ – moderate regression performance.

4 Diabetes Prediction - Classification

4.1 Python Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score,
    confusion_matrix

df = pd.read_csv('/content/drive/MyDrive/diabetes.csv')
X = df.drop('Outcome', axis=1)
y = df['Outcome']

X_scaled = StandardScaler().fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.25,
    random_state=42)

clf = RandomForestClassifier()
clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)
print(accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Listing 3: Random Forest for Diabetes Prediction

4.2 Results

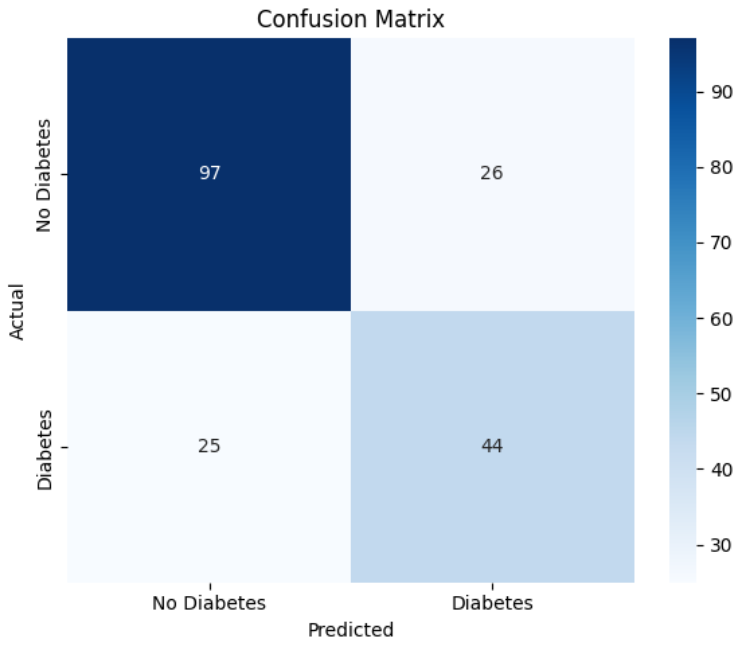

Figure 3: ROC Curve showing model performance with AUC = 0.80.

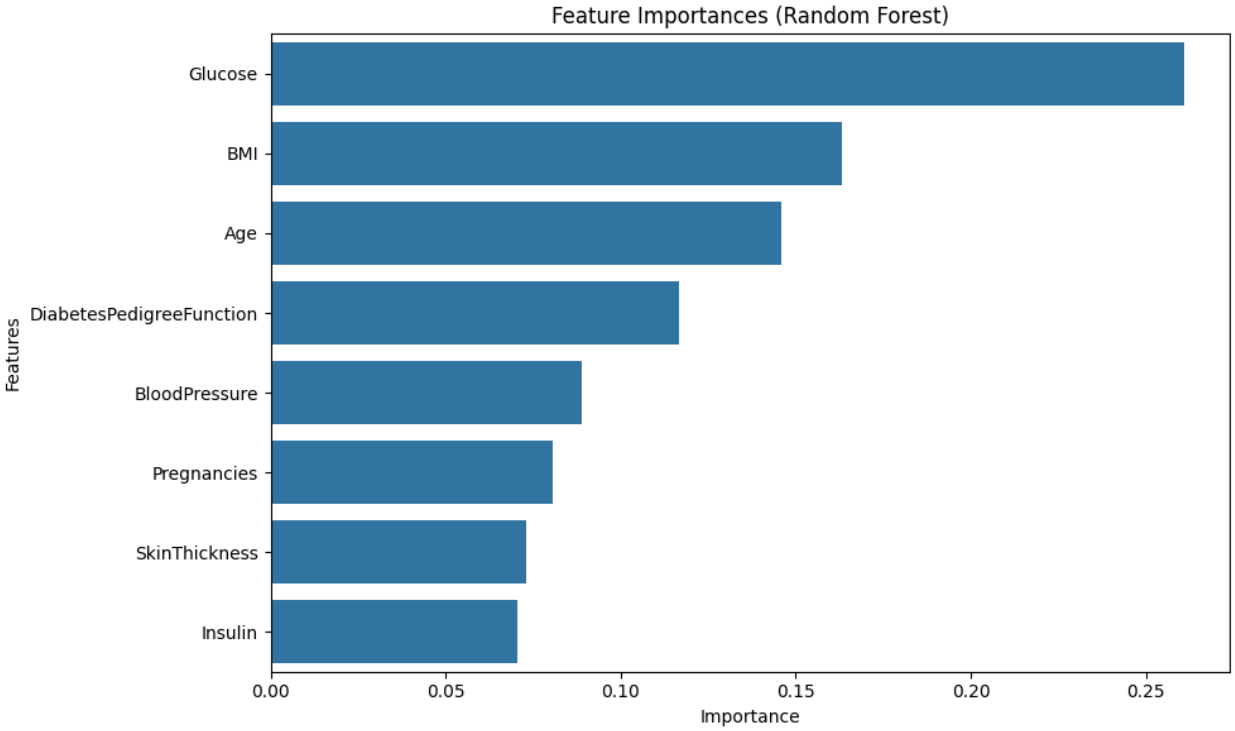

Figure 4: Feature importance of predictors using Random Forest.

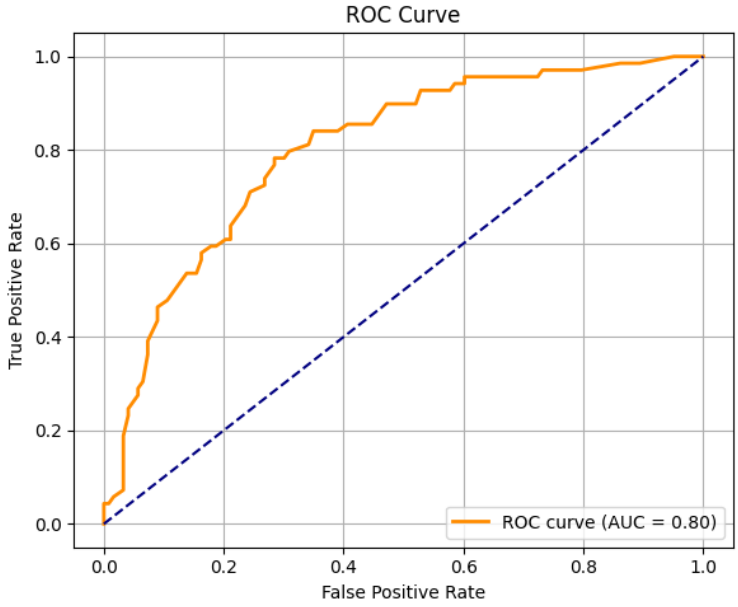

Figure 5: Confusion matrix for diabetes prediction model.

- Accuracy = 74%. Class 0 (No Diabetes) predicted better than Class 1 (Diabetes).

5 Spam Email Detection - Classification

5.1 Python Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report, accuracy_score

df = pd.read_csv("/content/drive/MyDrive/spambase.data", header=None)
X = df.iloc[:, :-1]
y = df.iloc[:, -1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=0)
model = GaussianNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print(accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Listing 4: Naive Bayes for Spam Classification

5.2 Results

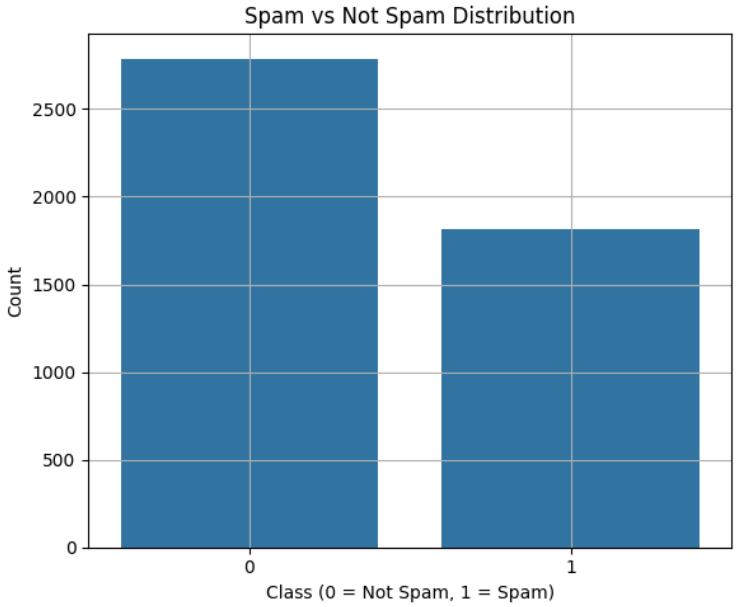

Figure 6: Distribution of Spam vs. Not Spam messages.

- Accuracy = 81%, with good recall for spam messages.

6 MNIST Digit Recognition

6.1 Python Code

```
from sklearn.datasets import load_digits
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score

digits = load_digits()
X, y = digits.data, digits.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
    random_state=42)
model = LogisticRegression(max_iter=3000)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
print(accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Listing 5: Logistic Regression on MNIST Digits

6.2 Results

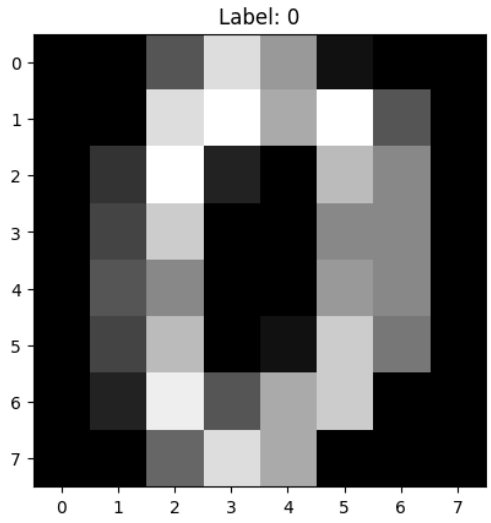

Figure 7: Sample MNIST digit image with label.

- Accuracy = 97%. Robust digit classification with Logistic Regression.

Table 1: Overview of ML experiments.

Dataset	**ML Task**	**Algorithm Used**
Iris	Classification	Logistic Regression
Loan Prediction	Regression	Linear Regression
Diabetes	Classification	Random Forest
Spam Emails	Classification	Naive Bayes
MNIST Digits	Classification	Logistic Regression

7 Discussion

8 Conclusion

The experiments demonstrated how different ML models perform on tasks such as diabetes prediction, spam detection, digit recognition, and loan prediction.

Key takeaways:

- Understood data preprocessing and feature selection.
- Learned to apply regression and classification models.
- Gained experience with evaluation metrics: Accuracy, Precision, Recall, F1-score, AUC, RMSE, R^2.
- Identified strengths and limitations of Logistic Regression, Naive Bayes, and Random Forest.