Final Project

Diabetes Classification Models Evaluation

1. Project Overview

Objective:

Evaluate and compare machine learning models for diabetes classification using the Pima Indians Diabetes dataset.

Models Implemented:

- Random Forest:
- K-Nearest Neighbors (KNN):
- GRU:

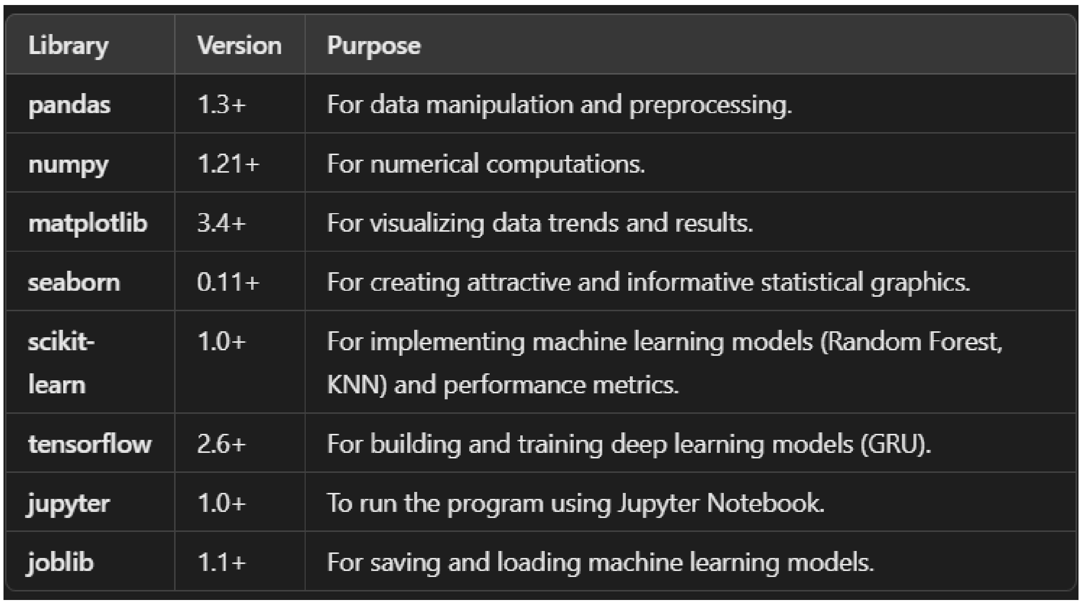
Repos: **https://github.com/shreyal-sharma/classification_model_evaluation**

2. Dataset

Source of the dataset: **Kaggle**

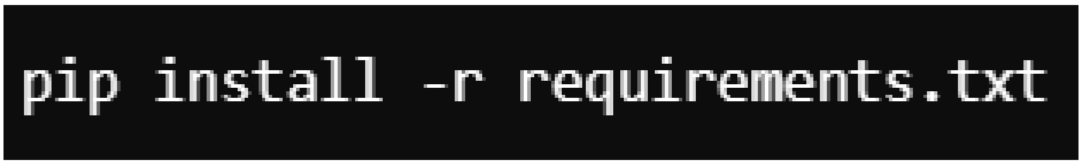
3. Software Requirements

- **Programming Language:** Python 3.x.
- **Required Packages:**

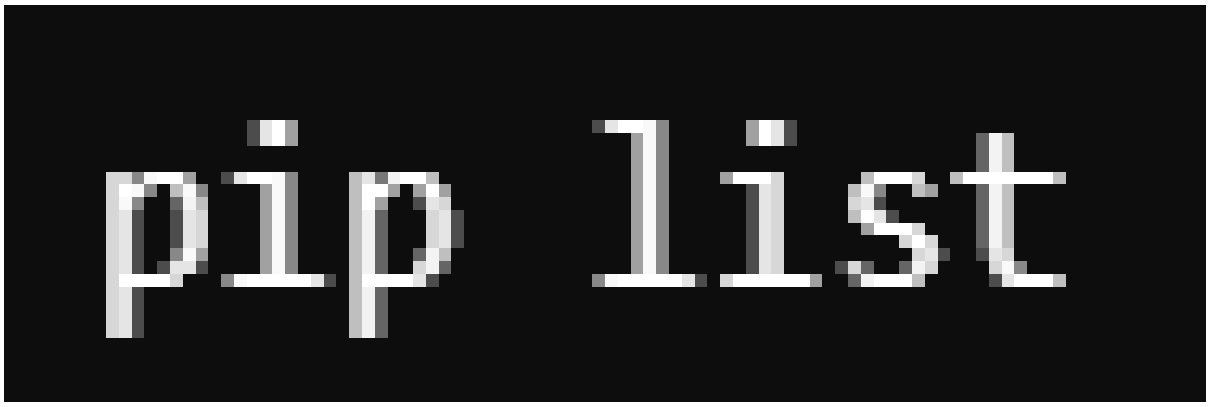


Library	Version	Purpose
pandas	1.3+	For data manipulation and preprocessing.
numpy	1.21+	For numerical computations.
matplotlib	3.4+	For visualizing data trends and results.
seaborn	0.11+	For creating attractive and informative statistical graphics.
scikit-learn	1.0+	For implementing machine learning models (Random Forest, KNN) and performance metrics.
tensorflow	2.6+	For building and training deep learning models (GRU).
jupyter	1.0+	To run the program using Jupyter Notebook.
joblib	1.1+	For saving and loading machine learning models.

- **Installing Required Libraries:**



```
pip install -r requirements.txt
```



```
pip list
```

4. How to Set Up the Project

Step-by-step guide to set up the environment and run the program:

1. Download ZIP
2. Extract the downloaded ZIP file and navigate to the project folder
3. Or Clone the Repository

5. Project Workflow

Data Preprocessing:

```
# Load the dataset
url = "dataset/pima-indians-diabetes-database.csv"  # Ensure correct path
data = pd.read_csv(url)

# Display dataset information
print(data.head())
print(f'Total number of records: {len(data)}')
print("Columns are: ", data.columns)
data.info()

# Check for missing values
if data.isnull().values.any():
    print("Dataset contains missing values.")
else:
    print("No missing values in the dataset.")
```

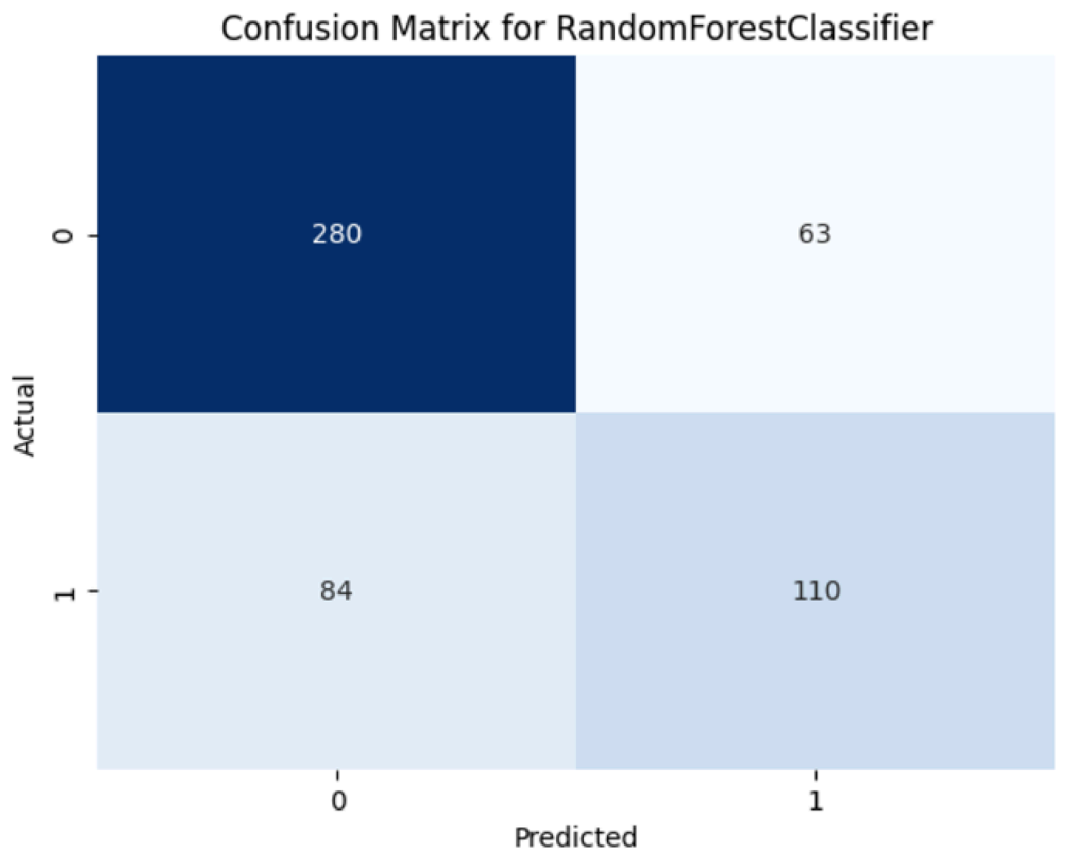
Models:

- Random Forest is an ensemble method that builds multiple decision trees and merges their outputs (averaging for regression, majority voting for classification).
- KNN is included to assess performance for distance-based learning approaches, especially for small or less complex datasets.
- GRU is a type of recurrent neural network (RNN) designed to capture temporal dependencies in sequential data. Unlike traditional RNNs, GRUs address the vanishing gradient problem with gating mechanisms, making them more efficient and effective for sequence modeling tasks.

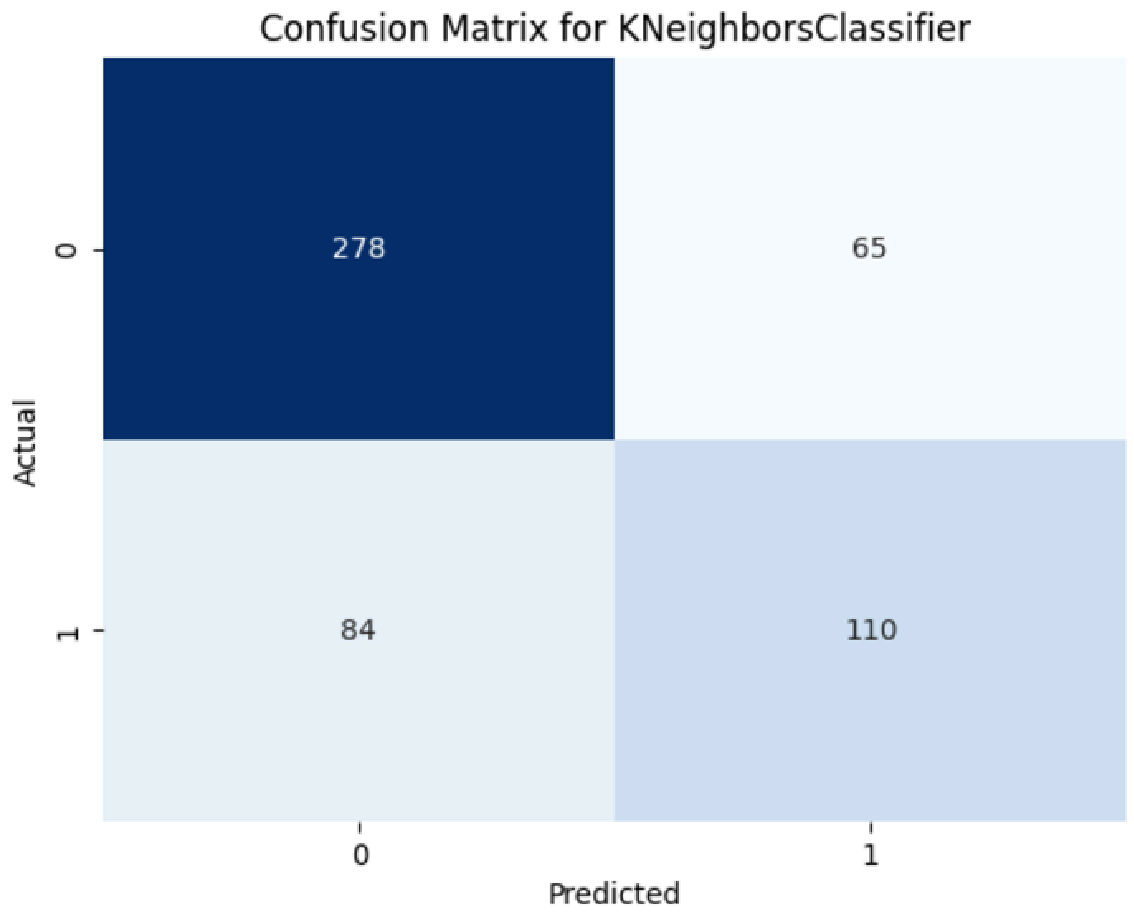
Results

Confusion matrix for the three models:

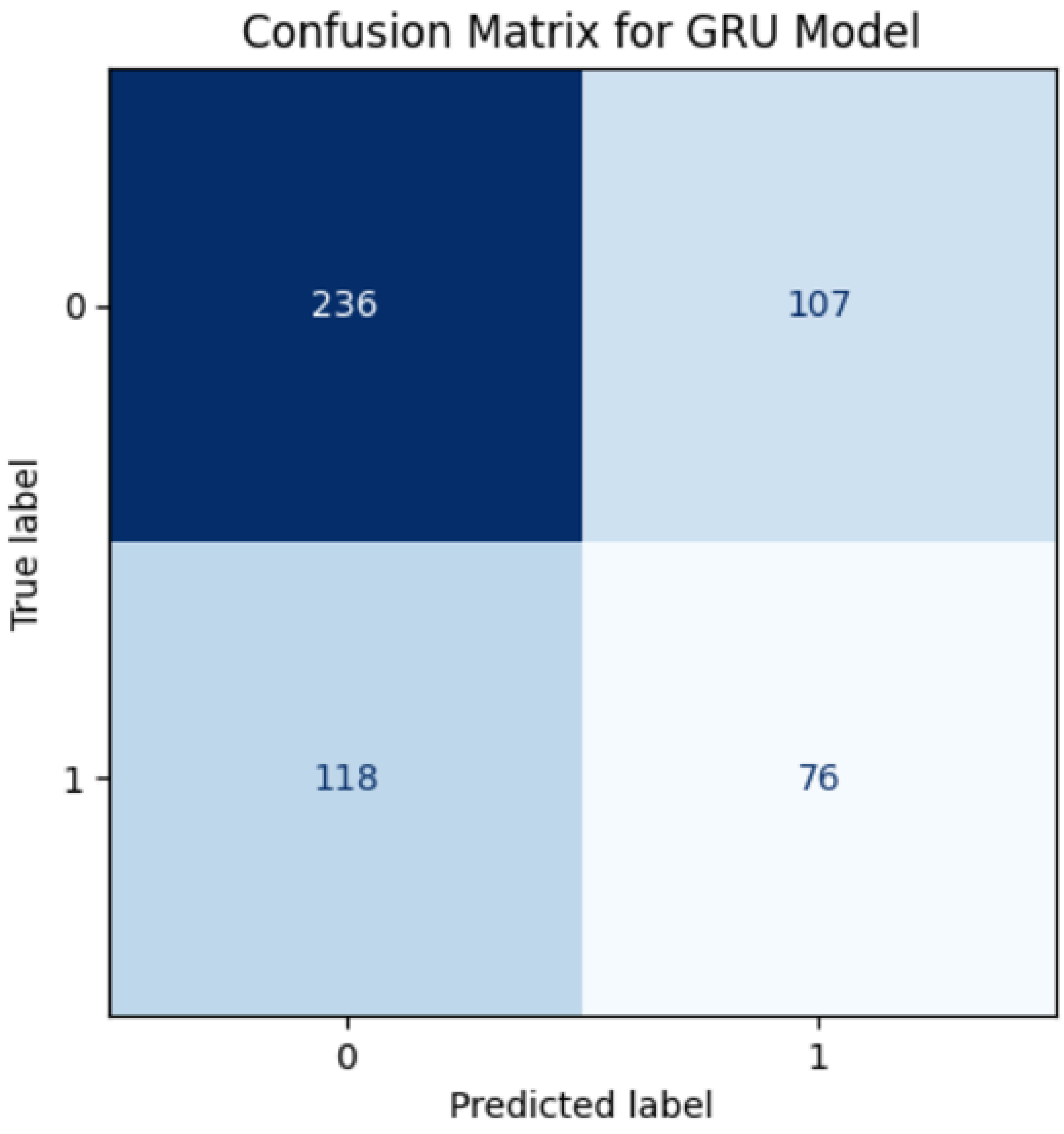
```
Performing 10-Fold Cross Validation for Random Forest...
```



```
Performing 10-Fold Cross Validation for KNN...
```



```
Confusion Matrix for GRU Model:
```



Metrix comparison

```
Model Comparison (Average Metrics per Model across 10 Folds):
```

	Model	Accuracy	Precision	Recall	F1-Score	TSS	HSS
0	Random Forest	0.726415	0.641096	0.567828	0.598322	0.383459	0.452830
1	KNN	0.722851	0.636645	0.572665	0.596494	0.384371	0.445702
2	GRU	0.581132	0.409342	0.400344	0.399125	0.085502	0.162264

Conclusion

Random Forest:

- **Accuracy:** 0.7264 (highest among the three models).

- **Precision:** 0.6411 (also the highest), indicating that when the model predicts a positive class (diabetes), it is more confident in its prediction than KNN and GRU.
- **Recall:** 0.5678, showing that the model correctly identifies a good portion of the actual positives, although still missing some.
- **F1-Score:** 0.5983, a balance between precision and recall, indicating decent performance on the imbalanced dataset.
- **TSS and HSS:** These values are also higher compared to KNN and GRU, suggesting the Random Forest is better at distinguishing between positive and negative cases and overall better predictive performance.

Which algorithm performs better and why?

Random Forest performs the best overall because:

1. Better Accuracy
2. Higher Precision and Recall
3. Balance between Precision and Recall
4. Higher TSS and HSS

Why did GRU perform poorly?

The GRU model is a powerful tool for sequential data, but it seems to be underperforming because:

- **Inappropriate Model Choice**
  - The GRU is designed for time-series or sequential data, and may not be the ideal choice for this tabular dataset.
- **Feature Representation**
  - The dataset is tabular, and deep learning models like GRU may not perform well unless properly preprocessed or tuned for tabular data.

The more effective model for this issue is **Random Forest**. Its improved accuracy and classification metrics performance is probably due to its ensemble nature and capacity to handle skewed datasets. KNN performs well as well, yielding results that are comparable to Random Forest's. However, Random Forest beats it on the majority of measures. The GRU model is not a good fit for this tabular dataset without significant adjustments or enhancements, even though it excels at sequential tasks.