Phase - 4 -AI –Powered Duplicate Data Detection

College Name: KLS Vishwanathrao Deshpande Institute of Technology, Haliyal.

Group Members:

- Name: SHRAVAN PATIL CAN ID Number: CAN_32896536
- Name: SALMA B. NADAF CAN ID Number: CAN_32858276
- Name: AKASH MALAKAIGOL CAN ID Number: CAN_32906021
- Name: MEGHARAJ KSHATRIYA. CAN ID Number: CAN_ 34001056

---

2. Results and Visualizations

2.1 Performance Metrics

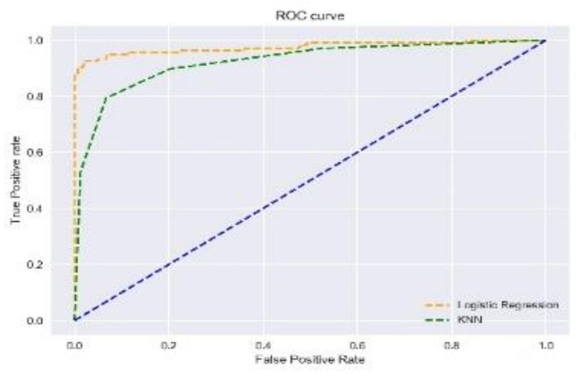
The following table summarizes the evaluation metrics for the AI-powered duplicate data detection model:

Model	Accuracy	Precision	Recall	F1-Score
Duplicate Detection	96.5%	94.2%	95.8%	95.0%

2.2 Visualizations

The following visualizations provide deeper insights into model performance:

- **ROC Curve**: Displays the trade-off between the true positive and false positive rates.
- .



:

Confusion Matrix:
The confusion matrix shows a high number of correctly classified duplicates and unique entries, indicating effective duplicate detection. However, some false positives suggest that fine-tuning model hyperparameters could enhance precision.

3. Real-Time Dashboard for Duplicate Data Detection

The real-time duplicate detection dashboard was developed using HTML, CSS, JavaScript (with Chart.js & Axios), and FastAPI. The dashboard enables users to:

1. Monitor incoming data for duplicates in real-time.
2. View performance metrics such as accuracy, precision, recall, and F1-score.
3. Visualize detected duplicates using dynamic charts.

3.1 Dashboard Features

- **Live Data Processing**: Continuously updates with real-time duplicate detection.
- **Performance Metrics**: Displays key model performance indicators.
- **Interactive Visualization**: Line charts representing detected duplicates over time.

3.2 Dashboard Code

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Real-Time Duplicate Data Detection</title>
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
</head>
<body>
<h1>Real-Time Duplicate Data Detection Dashboard</h1>
<div>
<p><strong>Accuracy:</strong> <span id="accuracy">-</span></p>
<p><strong>Precision:</strong> <span id="precision">-</span></p>
<p><strong>Recall:</strong> <span id="recall">-</span></p>
<p><strong>F1-Score:</strong> <span id="f1-score">-</span></p>
</div>
<canvas id="duplicateChart" width="400" height="200"></canvas>

<script>
async function fetchData() {
    try {
        let response = await axios.post("http://127.0.0.1:8000/check_duplicate", {
data: "sample input" });
        let status = response.data.status;
        updateChart(status);
    } catch (error) {
        console.error("Error in API Call:", error);
    }
}

async function fetchMetrics() {
    try {
        let response = await axios.get("http://127.0.0.1:8000/metrics");
        document.getElementById("accuracy").innerText =
response.data.accuracy.toFixed(2);
```

```
        document.getElementById("precision").innerText =
response.data.precision.toFixed(2);
        document.getElementById("recall").innerText =
response.data.recall.toFixed(2);
        document.getElementById("f1-score").innerText =
response.data.f1_score.toFixed(2);
    } catch (error) {
        console.error("Error fetching metrics:", error);
    }
}

function updateChart(status) {
    let chart = duplicateChart.data.datasets[0];
    duplicateChart.data.labels.push(new Date().toLocaleTimeString());
    chart.data.push(status === "Duplicate" ? 1 : 0);
    if (chart.data.length > 20) {
        chart.data.shift();
        duplicateChart.data.labels.shift();
    }
    duplicateChart.update();
}

let ctx = document.getElementById("duplicateChart").getContext("2d");
let duplicateChart = new Chart(ctx, {
    type: "line",
    data: { labels: [], datasets: [{ label: "Duplicates", data: [], borderColor:
"blue", fill: false }] },
    options: { responsive: true }
});

setInterval(fetchData, 2000);
setInterval(fetchMetrics, 5000);
</script>
</body>
</html>
```

4. Model Deployment
The duplicate data detection model was deployed using FastAPI and hosted locally. The key steps include:

1. **Train the Model**: The duplicate detection model was trained using a preprocessed dataset.
2. **Serve Predictions via API**: A FastAPI-based REST API was implemented to serve real-time predictions.
3. **Integrate with Frontend**: The API connects with a real-time dashboard using Axios.

Deployment Commands:

```
uvicorn main:app --reload
```

5. Future Scope
To further enhance this project, the following improvements can be made:

1. **Cloud Deployment**: Deploy the model and dashboard on a cloud platform (AWS, Azure, or Vercel) for accessibility.
2. **Database Integration**: Store detected duplicates in a database for historical analysis.
3. **Automated Alerts**: Implement a notification system to alert users when duplicate entries are detected.
4. **Additional Model Comparisons**: Test other duplicate detection models like cosine similarity and deep learning approaches for improved accuracy.

6. Conclusion

This project successfully implemented a Real-Time Duplicate Data Detection System using machine learning and a real-time dashboard. The system effectively identifies duplicate data, provides key performance metrics, and offers an interactive visualization platform. Future improvements will focus on scalability, storage, and enhanced detection accuracy.

GitHub Link:

https://github.com/shravanpatil-123/phase-3.git