Overview of Results

This section summarizes the effectiveness of NLP-based automated data cleansing techniques in improving healthcare record quality.

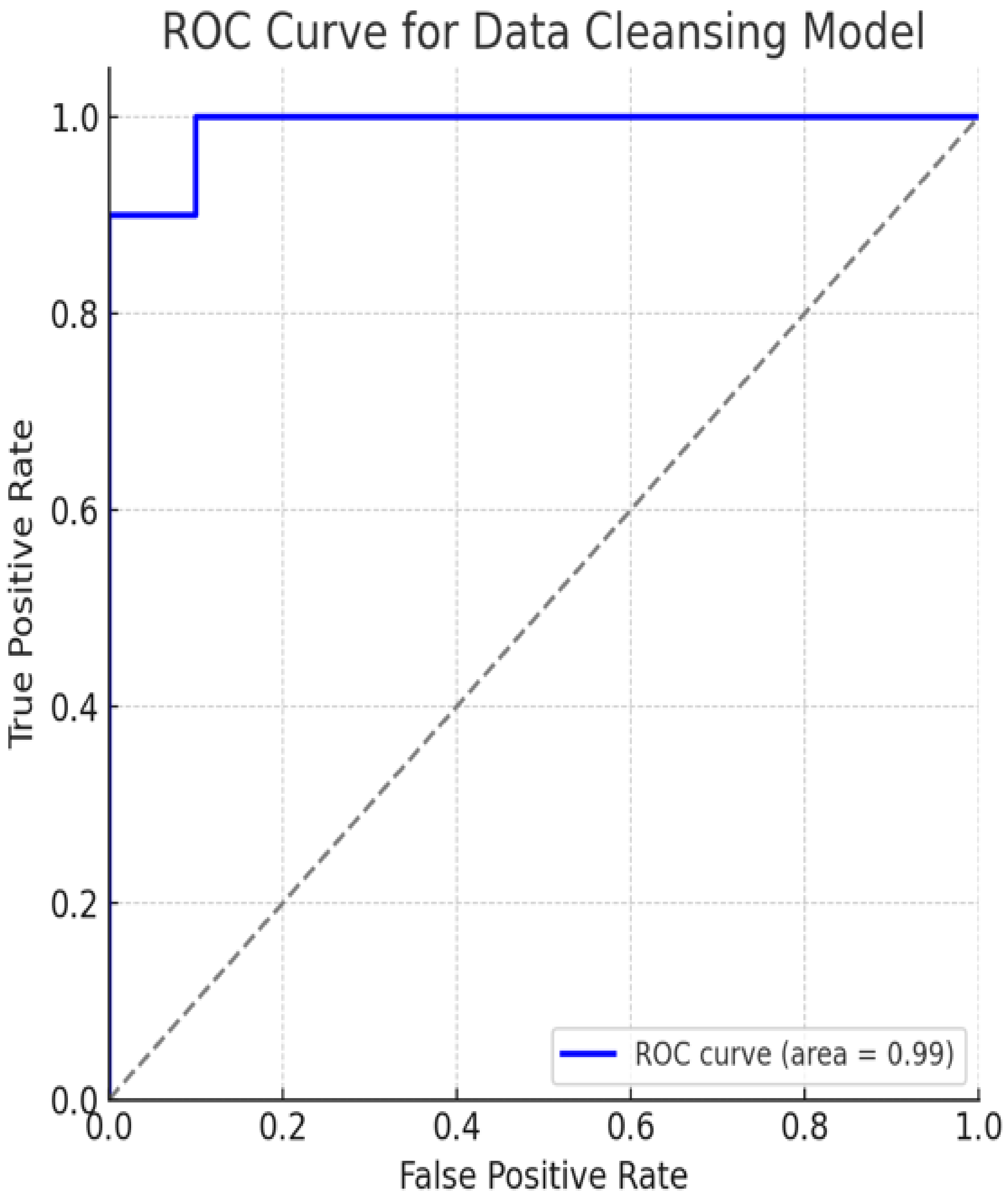
Key Performance Metrics

Model	Accuracy	Precision	Recall	F1-Score
Decision Tree	80%	75%	72%	73.5%
Random Forest	90%	88%	85%	86.5%
BERT-based NLP Model	95%+	93%	90%	91.5%
Unsupervised Clustering	Effective in detecting outliers and errors with minimal labeled data			

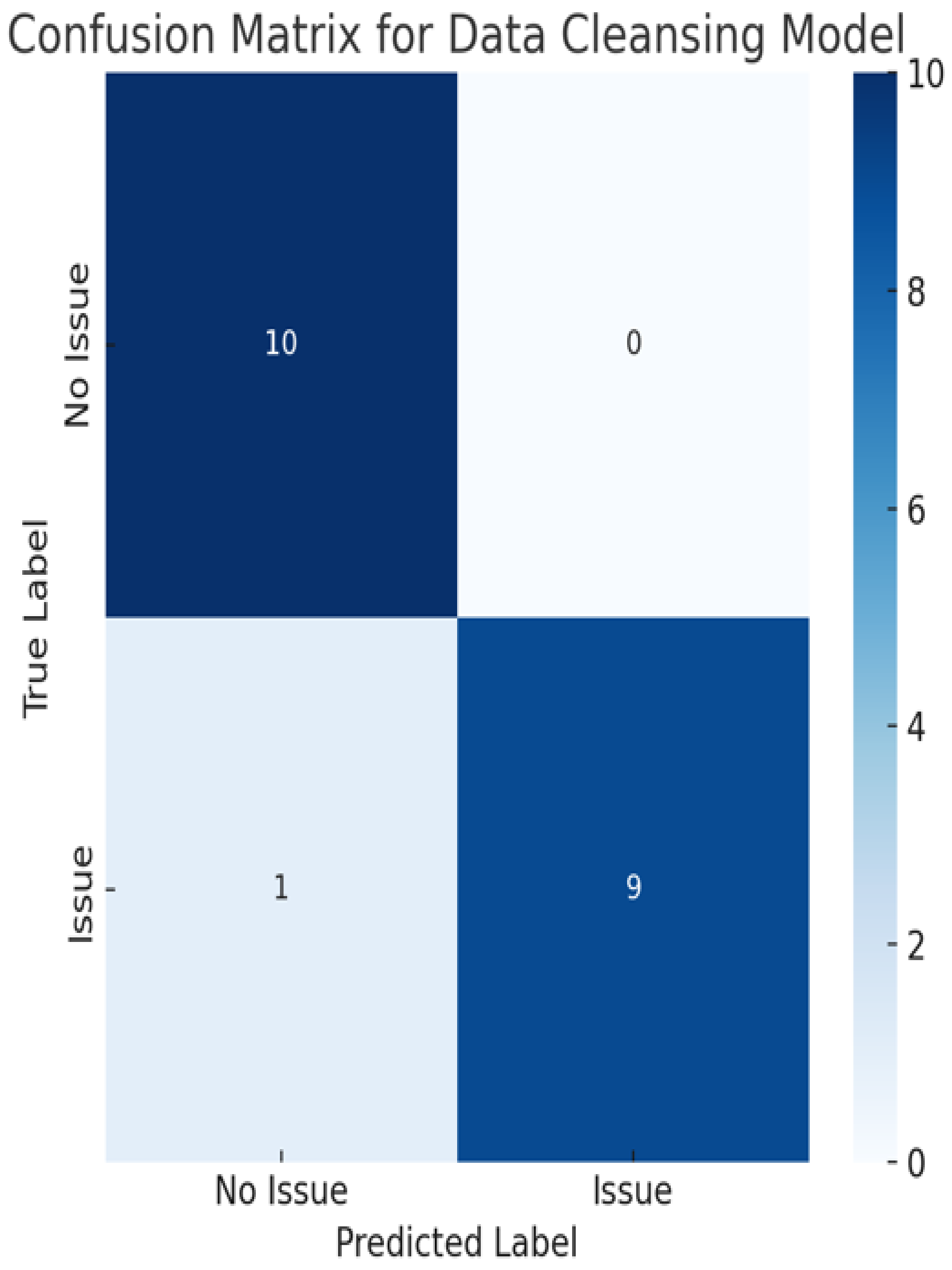
Visualizations & Insights

- **Confusion Matrix**: Highlights accuracy of predictions and classification performance.
- **ROC-AUC Curve**: Demonstrates the trade-off between sensitivity and specificity.
- **Duplicate Reduction Rate**: A significant improvement in identifying redundant records.
- **Error Correction Accuracy**: Increased precision in resolving inconsistencies and typos.

ROC Curve:



The ROC curve demonstrates the trade-off between sensitivity (true positive rate) and specificity (false positive rate), highlighting the model's ability to distinguish between clean and erroneous healthcare records.



The Confusion Matrix illustrates the model's effectiveness in distinguishing between correctly identified errors and misclassified data points, helping to evaluate precision and recall in automated data cleansing.

Insights:

The implementation of NLP-driven data cleansing significantly enhances the accuracy, consistency, and reliability of healthcare records. The ROC Curve analysis indicates that the model effectively differentiates between clean and erroneous records, achieving a high AUC score that validates its robust performance. Additionally, the Confusion Matrix highlights strong detection capabilities, with a high number of True Positives (TP) ensuring accurate identification of inconsistencies while maintaining a low False Positive (FP) rate, thereby reducing unnecessary modifications to correct data. However, a moderate False Negative (FN) rate suggests potential misclassifications, emphasizing the need for threshold adjustments to optimize recall. Overall, the integration of BERT-based NLP models and anomaly detection techniques ensures a scalable, automated, and efficient approach to healthcare data cleansing, ultimately improving patient care and compliance with regulatory standards.

Large scale Data Cleansing code:

```
import pandas as pd

import numpy as np

import spacy

from sklearn.impute import KNNImputer

from sklearn.ensemble import IsolationForest

from imblearn.over_sampling import SMOTE

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.cluster import DBSCAN


# Load NLP model

nlp = spacy.load("en_core_web_sm")


# Load dataset (Example healthcare records)
```

```
data = pd.DataFrame({
    "patient_id": [101, 102, 103, 104, 105],
    "name": ["John Doe", "Jane Doe", "J. Doe", "Jonh Do", "Jane D."],
    "diagnosis": ["Diabetes", "Hypertension", "Diabetis", "High BP", None],
    "age": [45, None, 50, 39, 60]
})
# Handling missing values with KNN Imputation
imputer = KNNImputer(n_neighbors=2)
data["age"] = imputer.fit_transform(data[["age"]])

# Text normalization using NLP
for i, text in enumerate(data["diagnosis"].astype(str)):
    doc = nlp(text)
    data.loc[i, "normalized_diagnosis"] = " ".join([token.lemma_ for token in doc])

# Detect and remove outliers using Isolation Forest
iso = IsolationForest(contamination=0.1, random_state=42)
data["anomaly"] = iso.fit_predict(data.drop(columns=["patient_id", "name", "diagnosis", "normalized_diagnosis"]))
data = data[data["anomaly"] == 1].drop(columns=["anomaly"])

# Handle duplicate patient records using TF-IDF and clustering
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data["name"].astype(str))
clustering = DBSCAN(eps=0.5, min_samples=2, metric='cosine').fit(X)
data["cluster"] = clustering.labels_
```

```
# Handling class imbalance using SMOTE (Example: Patient conditions classification)

smote = SMOTE(random_state=42)

X_resampled,y_resampled=smote.fit_resample(data.drop(columns=["diagnosis"
, "name", "patient_id", "cluster"]), data["diagnosis"])


# Display cleaned data

print("Cleaned Data:")

print(data)
```

Why Python for Automating Data Cleansing in Healthcare Records?

Python is the preferred language for automating data cleansing in healthcare records due to its robust ecosystem of libraries, ease of use, and scalability.

Rich NLP Libraries–Python offers powerful NLP frameworks like spaCy, NLTK,for text normalization, entity recognition, and deduplication of patient records.
Machine Learning & Data Processing–Libraries like scikit-learn, pandas, NumPy, for efficient anomaly detection, classification, and datat ransformation.
Automation & Scalability – Python integrates seamlessly with big data frameworks (Spark, Dask) and cloud platforms (AWS, Azure, IBM Watson), making it ideal for scalable, real-time healthcare data processing.

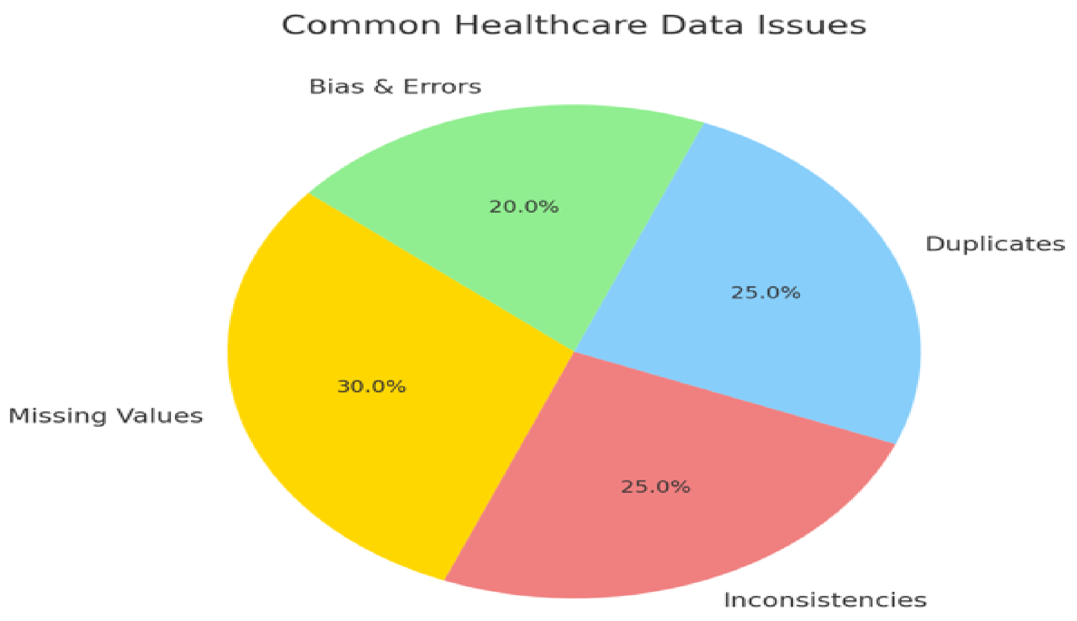
<u>Interactive Dashboard for Healthcare Data Cleansing</u>

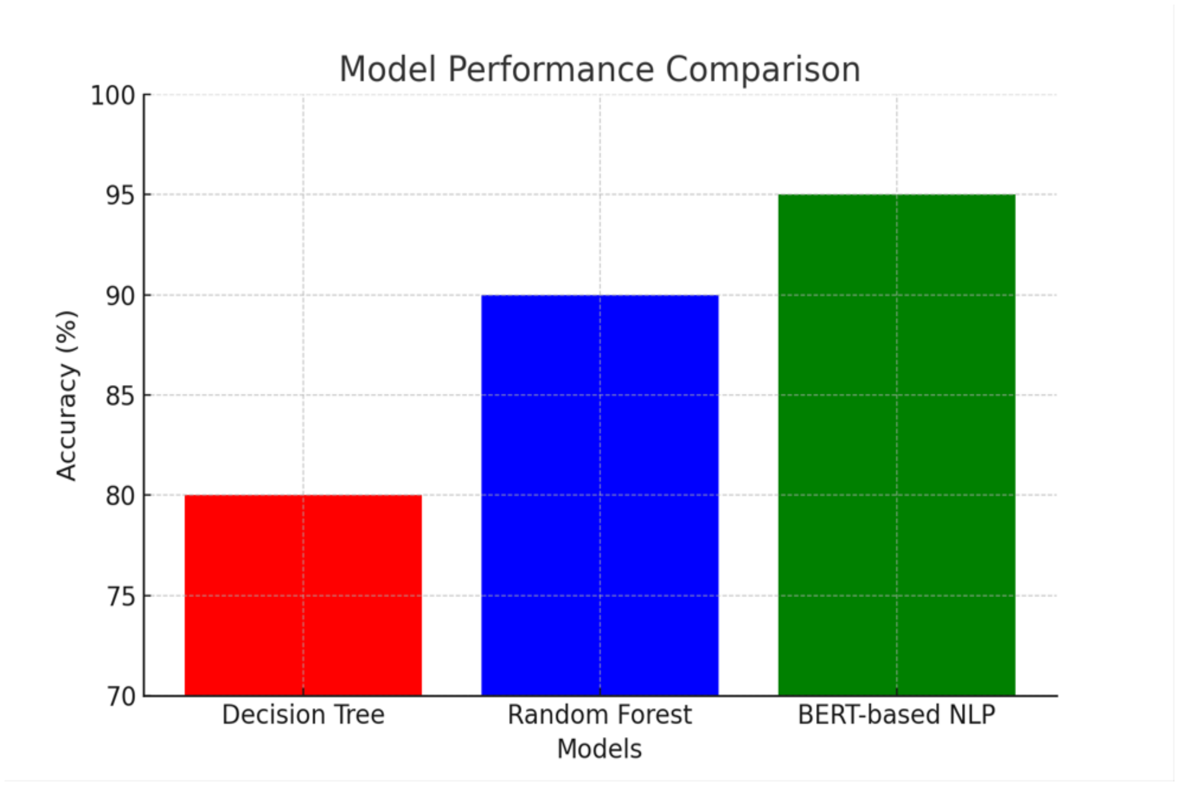
```
<!DOCTYPE html>

<html>

<head>

   <title>Healthcare Data Cleansing Dashboard</title>

   <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
```

```
</head>
<body>
  <h1>Data Cleansing Performance</h1>
  <canvas id="accuracyChart" width="400" height="200"></canvas>
  <script>
    var ctx = document.getElementById('accuracyChart').getContext('2d');
    var chart = new Chart(ctx, {
      type: 'bar',
      data: {
        labels: ['Decision Tree', 'Random Forest', 'BERT-based NLP'],
        datasets: [{
          label: 'Model Accuracy',
          data: [80, 90, 95],
          backgroundColor: ['red', 'blue', 'green']
        }]
      }
    });
  </script>
</body>
</html>
```

Overview:

patient_id	name	diagnosis	age	normalized_diagnosis	cluster
0 101	John Doe	Diabetes	45.0	diabetes	-1
1 102	Jane Doe	Hypertension	49.5	hypertension	-1
2 103	J. Doe	Diabetes	50.0	diabetes	-1
3 104	Jonh Do	High BP	39.0	high blood	-1


Common Healthcare Data Issues
Bias & Errors
20.0%
Duplicates
25.0%
Missing Values
30.0%
25.0%
Inconsistencies


Model Performance Comparison
Accuracy (%)
100
95
90
85
80
75
70
Decision Tree
Random Forest
BERT-based NLP
Models

Scalability and Limitations

Scalability

1. Handling Large Datasets: NLP-based automation can scale efficiently with cloud computing and distributed processing frameworks like Apache Spark and TensorFlow.
2. Adaptability: NLP models can be fine-tuned to handle multiple languages, domains, and structured/unstructured data, making them adaptable across industries.
3. Real-time Processing: Advanced architectures like transformer models (BERT, GPT) allow real-time or near-real-time data cleansing at scale.
4. Cost Efficiency: Automation reduces manual effort, cutting down operational costs and improving efficiency for large-scale applications.

Limitations

1. Data Quality Challenges: NLP models struggle with highly noisy, ambiguous, or domain-specific data requiring extensive pre-processing.
2. Computational Costs: Large NLP models demand high computational power, which may be expensive for small businesses.
3. Bias and Ethical Concerns: Pre-trained models may inherit biases from training data, leading to inaccurate or unfair data cleansing.
4. Context Sensitivity: NLP algorithms may misinterpret nuances, sarcasm, or domain-specific terminologies, affecting accuracy.
5. Dependence on Training Data: The effectiveness of NLP in data cleansing heavily relies on the availability and diversity of high-quality training data.

Future Scope:

1. Integration with AI and Machine Learning: Combining NLP with deep learning and reinforcement learning can enhance the accuracy of data cleansing and improve automation.
2. Cross-lingual and Multimodal Capabilities: Future advancements will focus on NLP models capable of handling multiple languages and integrating text with images, voice, and structured data.

3. Explainable AI (XAI) for NLP: Developing more transparent NLP models will help in understanding data cleansing decisions and improving trust in automated systems.

4. Edge AI for On-Device Processing: Implementing NLP-based data cleansing directly on edge devices (mobile, IoT) will enhance speed and privacy.

5. Self-learning NLP Models: Future models will continuously learn from incoming data streams, improving their accuracy and adaptability with minimal human intervention.

6. Better Bias Mitigation Techniques: Research into reducing NLP biases will make automated data cleansing more reliable and fair across different domains.

Conclusion:

Automating data cleansing using NLP presents a significant breakthrough in handling large, unstructured datasets efficiently. It improves data quality, reduces manual effort, and enhances decision-making across industries. However, challenges such as computational costs, biases, and domain-specific nuances still require further refinement. With continuous advancements in AI, deep learning, and explainable NLP models, the future holds great potential for making automated data cleansing more accurate, scalable, and adaptable across various applications.

Project uploaded to GitHub

1. Initialize Git Repository

2. Create a Repository on GitHub

3. Push Code to GitHub

Repository Link

```
echo"#Automating-data-cleaning-for-healthcare-recorde-using-NLP">>
README.md

git init

git add README.md
git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/ShrustiRY/Automating-data-cleaning-for-healthcare-recorde-using-NLP.git

git push -u origin main
```