Solution Architecture for cleaning Healthcare dataset

1. Libraries and Data Initialization

- **Libraries:**
    - pandas: For creating and managing the structured dataset.
    - matplotlib.pyplot and seaborn: For data visualization.
    - sklearn: For anomaly detection (IsolationForest) and feature scaling (StandardScaler).
    - nltk: For natural language processing tasks (tokenization).
- **Data Initialization:**

data = pd.DataFrame({

'age': [25, 30, 45, 40, 50],

'lab_test_result': [5.1, 4.9, 6.0, 5.8, 7.1],

'diagnosis': [0, 1, 0, 1, 0]

})

    - Creates a small sample dataset containing:
        - age: Age of patients.
        - lab_test_result: A numeric result from some lab test.
        - diagnosis: Encoded medical diagnosis (e.g., 0 = negative, 1 = positive).

2. Data Visualization

Histogram

sns.histplot(data['age'], kde=True, bins=5)

plt.title('Age Distribution')

plt.xlabel('Age')

plt.ylabel('Frequency')

plt.show()

- A histogram is generated to visualize the distribution of patient ages.
    - kde=True: Adds a Kernel Density Estimate curve over the histogram.
    - bins=5: Sets the number of bins in the histogram.
    - The title, xlabel, and ylabel add context to the chart.

3. Anomaly Detection

Feature Scaling

scaler = StandardScaler()

data_scaled = scaler.fit_transform(data[['age', 'lab_test_result']])

- Scales the numeric features (age and lab_test_result) to have a mean of 0 and standard deviation of 1. This step is essential for algorithms like Isolation Forest that are sensitive to feature magnitudes.

Isolation Forest

iso_forest = IsolationForest(contamination=0.2, random_state=42)

data['anomaly'] = iso_forest.fit_predict(data_scaled)

- Anomaly detection is performed using Isolation Forest:
    - contamination=0.2: Specifies that 20% of the data points are expected to be anomalies.
    - fit_predict: Fits the model to the data and assigns anomaly labels:
        - 1 = Normal.
        - -1 = Anomaly.

Mapping Anomalies

data['anomaly_label'] = data['anomaly'].map({1: 'Normal', -1: 'Anomaly'})

print("Anomaly Detection Results:")

print(data[['age', 'lab_test_result', 'anomaly_label']])

- Converts the numeric anomaly labels (1 and -1) into human-readable labels (Normal and Anomaly).
- Displays the age, lab test result, and anomaly status for each patient.

4. Basic NLP

Tokenization

```
nltk.download('punkt', quiet=True)

clinical_note = "Patient shows elevated blood pressure."

words = word_tokenize(clinical_note)

print("Tokenized Words:", words)
```

- Downloads the punkt tokenizer from NLTK.
- Tokenizes the clinical_note into individual words:
    - Input: "Patient shows elevated blood pressure."
    - Output: ['Patient', 'shows', 'elevated', 'blood', 'pressure', '.']

Key Outputs

1. **Age Distribution Histogram:**
    - Visualizes the spread of patient ages in the dataset.
2. **Anomaly Detection Results:**
    - Displays which data points are classified as anomalies or normal based on age and lab test results.
3. **Tokenized Words:**
    - Splits the clinical note into individual words and punctuation.