Phase 1: Data Cleaning

1. Data Loading and Initial Exploration

Step Taken:

- The training datasets (Train_Beneficiarydata.csv, Train_Inpatientdata.csv, Train_Outpatientdata.csv, and Train.csv) were loaded into the environment.
- Data shapes and first few rows were printed for a quick inspection of the loaded data.
- The **Labels** dataset includes the target variable PotentialFraud, and an analysis of its distribution was performed.
- **Missing values** were checked in the **Beneficiary**, **Inpatient**, and **Outpatient** datasets.

Outcome:

- All training datasets were successfully loaded. The dataset's PotentialFraud target variable showed an imbalanced distribution, with fewer fraudulent providers.

---

2. Data Preprocessing and Cleaning

Step Taken:

- **Beneficiary Data:**
  - Dates in DOB and DOD columns were converted to the datetime format.
  - **IsDeceased** and **Age** columns were created, filling missing values with the median.
- **Inpatient Data:**
  - Missing values in physician names, diagnosis and procedure codes were filled with placeholders.
  - **Admission/Discharge dates** were filled with frequent values or 'Unknown'.
  - Missing numeric values were filled with zeros.
- **Outpatient Data:**
  - Similar cleaning steps were applied to the outpatient dataset, including filling missing values for physician names, diagnosis codes, procedure codes, and claim dates.

Outcome:

- All datasets were cleaned, with missing values handled appropriately.

---

3. Verification and Final Cleaning Checks

Step Taken:

- The datasets were checked for any remaining missing values after cleaning.
- The **Fraud Distribution** check confirmed that the dataset was imbalanced, with fewer fraudulent providers.

Outcome:

- No missing values remained after the cleaning process. The imbalanced distribution of fraudulent providers was acknowledged early on.

---

4. Feature Engineering

Step Taken:

- **Combining Claims Data:** A new feature ClaimType was created to distinguish between Inpatient and Outpatient claims.
- **Provider-Level Features:** Aggregated features for each provider were created, such as total claims, unique patients, total reimbursement amounts, and more.
- **Patient-Level Features:** Features related to patient behavior, like age, chronic conditions, and annual reimbursement amounts, were created.
- **Final Dataset:** The provider-level and patient-level features were combined, and the target variable Fraud was created as a binary indicator (1 for fraud, 0 for non-fraud).

Outcome:

- A comprehensive final dataset with both provider-level and patient-level features was created, ready for model training.

---

5. Data Quality Check

Step Taken:

- A final check for any remaining missing values and infinity values was performed.
- **Feature Distribution** was analyzed to ensure consistency across key features like TotalClaims, AvgReimbursed, and UniquePatients.

Outcome:

- The dataset was confirmed to be clean and ready for model training.

---

6. Data Visualization

Step Taken:

- **Correlation Heatmap:** A heatmap was generated to visualize correlations between features.

- **Fraud vs. Non-Fraud Feature Comparison:** Box plots, violin plots, and histograms were used to visualize the distribution of key features like TotalClaims, AvgReimbursed, and UniquePatients in relation to fraud status.

Outcome:

- Visualizations helped understand the relationships between features and fraud status, highlighting differences between fraud and non-fraud providers.

---

7. Saving Processed Data

Step Taken:

- The final cleaned dataset was saved as train_processed.csv for further use in modeling.

Outcome:

- The processed data was successfully saved, with details about its shape and size.

---

8. Feature List

Step Taken:

- A list of all features used in the dataset, excluding the target variable and identifiers (Fraud, Provider, PotentialFraud), was compiled.

Outcome:

- A comprehensive list of features was created to document all variables used for modeling.

Phase 2: Modeling and Evaluation

2.1 Data Splitting and SMOTE Application

Step Taken:

- The dataset was split into training and testing sets using **stratified train-test split**, ensuring that the distribution of fraud was maintained.
- **SMOTE** was applied to the training data to address the class imbalance and enhance the model's ability to detect fraudulent providers.

Outcome:

- SMOTE successfully increased the number of training samples, while the test set remained unaffected by SMOTE.

2.2 Model Selection

Step Taken:

- Models chosen for evaluation included **Logistic Regression**, **Decision Tree**, **Random Forest**, and **Gradient Boosting**.

Rationale:

- These models were selected to capture different aspects of the dataset, from linear relationships (Logistic Regression) to non-linear patterns (Decision Tree, Random Forest, Gradient Boosting).

2.3 Model Training and Hyperparameter Tuning

Step Taken:

- Models were trained using the resampled training data, and **hyperparameter tuning** was performed using **GridSearchCV** for **Random Forest** and **Gradient Boosting**.

Outcome:

- The **best parameters** for **Random Forest** and **Gradient Boosting** were found, resulting in improved performance.

2.4 Model Evaluation

Step Taken:

- Models were evaluated using **classification reports**, **ROC-AUC**, **PR-AUC** scores, and **confusion matrices**.

Outcome:

- **Random Forest** performed the best with **ROC-AUC = 0.95** and **PR-AUC = 0.92**, making it the top choice for fraud detection.

2.5 Model Comparison

Step Taken:

- A **comparison table** was created to compare the performance of all models based on **ROC-AUC** and **PR-AUC**scores.

```
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.89      0.87       981
           1       0.88      0.85      0.87       981

    accuracy                           0.87      1962
   macro avg       0.87      0.87      0.87      1962
weighted avg       0.87      0.87      0.87      1962

Logistic Regression ROC-AUC: 0.8705402650356779
Logistic Regression PR-AUC: 0.82749968514769
```

```
Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.90      0.91       981
           1       0.90      0.92      0.91       981

    accuracy                           0.91      1962
   macro avg       0.91      0.91      0.91      1962
weighted avg       0.91      0.91      0.91      1962

Decision Tree ROC-AUC: 0.9092762487257899
Decision Tree PR-AUC: 0.86879837797269
```

```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.93      0.95       981
           1       0.94      0.96      0.95       981

    accuracy                           0.95      1962
   macro avg       0.95      0.95      0.95      1962
weighted avg       0.95      0.95      0.95      1962

Random Forest ROC-AUC: 0.946992864424057
Random Forest PR-AUC: 0.918140296213964
```

```
Gradient Boosting Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.89      0.91       981
           1       0.89      0.93      0.91       981

    accuracy                           0.91      1962
   macro avg       0.91      0.91      0.91      1962
weighted avg       0.91      0.91      0.91      1962

Gradient Boosting ROC-AUC: 0.9077471967380223
Gradient Boosting PR-AUC: 0.8640886069104738
```

```
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.89      0.87       981
           1       0.88      0.85      0.87       981

    accuracy                           0.87      1962
   macro avg       0.87      0.87      0.87      1962
weighted avg       0.87      0.87      0.87      1962

Logistic Regression ROC-AUC: 0.8705402650356779
Logistic Regression PR-AUC: 0.82749968514769
```

Outcome:

- **Tuned Random Forest** emerged as the best model.

2.6 Visualizations

Step Taken:

- **ROC-AUC** and **Precision-Recall** curves were plotted for **Random Forest**.
- **Confusion matrices** were visualized for each model.

Outcome:

- Visualizations confirmed that **Random Forest** had a strong ability to distinguish between fraudulent and non-fraudulent providers.

2.7 Error Analysis

Step Taken:

- **False positives** and **false negatives** were analyzed to assess model errors and their implications for fraud detection.

Outcome:

- **Random Forest** showed a balanced performance, with lower false negatives compared to other models.

2.8 Conclusion

Best Performing Model:

- **Tuned Random Forest** with the highest **ROC-AUC** and **PR-AUC** scores was selected as the best model.
-

```
Best Random Forest Parameters: {'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
Tuned Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.93      0.95       981
           1       0.94      0.96      0.95       981

    accuracy                           0.95      1962
   macro avg       0.95      0.95      0.95      1962
weighted avg       0.95      0.95      0.95      1962

Tuned Random Forest ROC-AUC: 0.9475025484199795
Tuned Random Forest PR-AUC: 0.9190332014032014
```

Final Model Deployment:

- The final model was saved as final_model.pkl for deployment.

Phase 3: Model Evaluation and Threshold Analysis

3.1 Model and Scaler Loading

Step Taken:

- The saved model (final_model.pkl) and scaler (scaler.pkl) were loaded for making predictions on the test data.

3.2 Loading and Scaling Test Data

Step Taken:

- The **test data** and **true labels** were loaded and scaled using the previously trained scaler.

3.3 Making Predictions

Step Taken:

- **Predictions** were made using the scaled test data, including both **default** (0.5) and **probabilistic** predictions.

3.4 Threshold Analysis

Step Taken:

- Performance was evaluated at various thresholds (0.50, 0.30, 0.25, etc.) to find the optimal balance between precision and recall.

Outcome:

- Threshold **0.25** was selected as the best threshold based on precision-recall trade-offs.

3.5 Confusion Matrix for Chosen Threshold

Step Taken:

- A confusion matrix was generated for the threshold **0.25** to visualize the model's performance.

3.6 ROC Curve

Step Taken:

- The **ROC curve** was plotted to visualize the true positive rate vs. false positive rate.

3.7 Precision-Recall Curve

Step Taken:

- The **Precision-Recall curve** was plotted to assess precision-recall trade-offs at different thresholds.

3.8 Error Analysis: False Positives & False Negatives

Step Taken:

- **False positives** and **false negatives** were analyzed, with the top 3 false positives and false negatives displayed.

3.9 Conclusion

Best Threshold Selection:

- **Threshold 0.25** was chosen as the optimal threshold.

Model Performance:

- The **Random Forest** model showed strong performance, and the threshold adjustment helped balance precision and recall effectively.