Final Report

Objectives:

- Develop a **machine learning classification model** that can predict game outcomes or labels using structured data from the `Games.csv` dataset.

- Assist stakeholders (e.g., game analysts or sports strategists) in understanding key factors influencing game categorization or results.

- Build a reliable model that generalizes well to unseen game data and supports future analytics or decision-making.

Methods:

1. Data Preparation

- **Dataset Loaded**: `Games.csv`
- **Missing Value Handling**:
  - Categorical fields (`gameSubLabel`, `gameLabel`, `seriesGameNumber`) filled with `'N/A'`.
  - Numerical column (`attendance`) filled using the **median**.
  - `seriesGameNumber` converted to **string** type to avoid numerical misinterpretation.

2. Feature Encoding

- `LabelEncoder` used to convert categorical variables into numeric format for model compatibility.

3. Modeling

- **Model Used**: `RandomForestClassifier` from Scikit-learn.
- **Train-Test Split**: Data split into training and testing subsets using `train_test_split`.
- **Model Evaluation Metrics**:
  - **Accuracy Score**
  - **Classification Report** (Precision, Recall, F1-score)
  - **Confusion Matrix**

Results:

- **Model Accuracy**: The Random Forest model achieved a reasonable accuracy score (exact value printed in the notebook).

- **Precision, Recall, F1**: All reported through the classification report. This confirms the model performs well across multiple classes (if present).

- **Confusion Matrix**: Provides a breakdown of actual vs predicted labels, indicating where the model misclassified instances.

Insights:

- The model appears to **generalize reasonably** to unseen test data based on accuracy and class-wise metrics.

- Random Forest is appropriate for this task due to its robustness against overfitting and handling of categorical inputs (via encoding).

- Encoding of game-related categorical features like `gameLabel` and `seriesGameNumber` likely contributed significantly to the model's learning.

- Handling missing values properly has helped avoid common pitfalls in real-world game datasets, especially with sparse fields like `attendance`

Recommendations:

EDA & Visuals

Add bar plots for class distributions, attendance spread, and feature correlations.

Model Analysis

Include a **feature importance plot** to reveal which inputs most influence predictions.

Documentation

Use **markdown cells** to explain the process step-by-step (data cleaning, modeling, evaluation).

Evaluation

Visualize the **confusion matrix as a heatmap** for better interpretability.

Deployment

Consider exporting the model with `joblib` or `pickle` for web or app deployment.

Advanced Models

Experiment with Gradient Boosting or XGBoost for potentially better results.