DIVISION OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

26AI2003 STATISTICAL PROGRAMMING FOR CLINICAL DATA ANALYTICS RECORD

Experiment No. 2
Date: 17/08/2026
Title / Aim: *Data Cleaning using Python, R, and SAS to produce a clean, analysis-ready dataset.*

1. Objective

To systematically identify, quantify, and treat missing clinical observations via baseline feature median array imputations; detect and eliminate exact row duplications along with conflicting patient identifier sequences; and correct, standardize, and scrub string text mutations or invalid biometric outlier distributions to yield an audited, pristine, clinical-grade analysis dataset.

2. Tools / Software Used

☒ SAS (Base SAS) ☒ R ☒ Python

3. Procedure / Methodology

1. Ingestion & Quality Audit: Load 'clinical_trial_data_raw.csv' across environments (Python pandas, R base/readr, SAS PROC IMPORT). Execute high-level structural scans via df.info(), colSums(is.na(df)), and PROC MEANS NMISS to quantify data fragmentation vectors.
2. Categorical Realignment: Detect mismatched case string elements within variable boundaries (e.g., 'SEX' featuring F, f, M, m, MALE, FEMALE variants; 'ARM' with casing discrepancies and extra spaces). Apply mapping dictionaries and serialization functions (.str.strip().str.upper()) to conform values to exact dictionary arrays.
3. Outlier Extraction: Scan biometric arrays for extreme or physically impossible data entries (e.g., 'AGE' records showing <= 0 or 999 values). Recode these anomalies explicitly to null variables (NaN) to prevent downstream mathematical bias.
4. Structural Consolidation: Identify and purge exact duplicate rows using df.drop_duplicates() or PROC SORT NODUPRECS. Resolve identity clashes by isolating conflicting patient identifiers (USUBJID) and executing first-occurrence validation retention rules.
5. Imputation Phase: Extract median profiles for all fragmented numerical metric distributions (AGE, WEIGHT_KG, BMI, GLUCOSE_MGDL, CHOLESTEROL_MGDL) and populate baseline missing cells. Flag string exclusions (SITEID) and overwrite with a standardized 'Unknown' category token before sealing the dataset.

4. Program / Code (Python Core Script Implementation)

```
import pandas as pd import
numpy as np

# 1. Load raw dataset and establish properties df=
pd.read_csv('clinical_trial_data_raw.csv')
print('Raw Rows:', df.shape[0], 'Columns:', df.shape[1]) print(df.isnull().sum())

# 2. Standardize inconsistent categorical text data
df['SEX'] = df['SEX'].astype('string').str.strip().str.upper()
df['SEX'] = df['SEX'].replace({'MALE': 'M', 'FEMALE': 'F'})
```

```
df['ARM'] = df['ARM'].astype('string').str.strip().str.upper() df['ARM'] =
df['ARM'].replace({
    'DRUG 10MG': 'Drug 10mg', 'DRUG10MG': 'Drug 10mg',
    'DRUG 20MG': 'Drug 20mg', 'DRUG20MG': 'Drug 20mg', 'PLACEBO': 'Placebo'
})

# 3. Identify and isolate invalid biometric age records invalid_age = (df['AGE'] <= 0) |
(df['AGE'] > 100) df.loc[invalid_age, 'AGE'] = np.nan

# 4. Standardize date parameters
df['ENROLL_DATE'] = pd.to_datetime(df['ENROLL_DATE'], format='mixed', errors='coerce') df['ENROLL_DATE'] =
df['ENROLL_DATE'].dt.strftime('%Y-%m-%d')

# 5. Deduping structural observations df =
df.drop_duplicates()
df = df.drop_duplicates(subset='USUBJID', keep='first')

# 6. Multi-column numerical median distribution imputation numeric_cols =
df.select_dtypes(include=np.number).columns for col in numeric_cols:
    if df[col].isnull().sum() > 0:
        df[col] = df[col].fillna(df[col].median())

# 7. Impute missing categorical string tracking metric df['SITEID'] =
df['SITEID'].fillna('Unknown')

# 8. Document validated tidy results df.to_csv('clinical_trial_data_clean.csv',
index=False) print('Final Verified Shape:', df.shape)
```

5. Output / Results (Data Processing Verification Profiles)

Table 5.1: Pipeline Structural Metrics & Deduplication Audits

Data Cleaning Checkpoint Metrics	Pre-Cleaned State Profile	Post-Cleaned Clean State
Total Raw Pipeline Data Rows Ingested	193 rows	180 rows (Consolidated)
Exact Row Multiplicity Counts Purged	8 rows detected	0 rows remaining
Conflicting Identity Keys Purged (USUBJID)	5 rows handled	0 identities clashed
Categorical Noise Vectors Found (SEX / ARM)	High Variance (Casing/Spaces)	Standardized Vector Arrays

Outlier Anomalies Cleaned (AGE: 0, 999, -5)	5 Invalid Records Present	All Recoded & Imputed

Table 5.2: Missing Value Quantification & Target Vector Imputations

Target Variable	Missing Counts	Fractional Impact	Resolution Imputation Base Value
AGE	4 records	2.22%	Median Imputation = 49.0 years
WEIGHT_KG	7 records	3.89%	Median Imputation = 70.7 kg
BMI	7 records	3.89%	Median Imputation = 26.1 kg/m²
GLUCOSE_MGDL	9 records	5.00%	Median Imputation = 95.3 mg/dL
CHOLESTEROL_MGDL	5 records	2.78%	Median Imputation = 197.2 mg/dL
SITEID	3 records	1.67%	Categorical Imputation = 'Unknown'

Table 5.3: Core Categorical Frequency Maps Post-Scrubbing

Scrubbed Tracking Domain	Categorical Code Groupings	Final Consolidated Frequencies
SEX Demographics	F (Female)	96 Patients
	M (Male)	84 Patients

Clinical Treatment Arms	Drug 20mg	68 Patients

	Drug 10mg	60 Patients
	Placebo	52 Patients

```
df = pd. read_csv( "clinical_trial_data_raw. cs.v"}

print('"RaL'lildataset loaded successfully ")
print("Rm.s:", df.shape[0])
print("Columns:", df.shape[l])
URK23CS7040
```

RaOIdataset loaded successfully.
Ro111s.: 193
Columns: 19

```
print("ARl1 values before cleaning:")
print(df["ARM"].value_counts(dropna=False))
URK23CS7040
```

ARM valuesbeforecleaning:
ARM

Drug 10mg	40
Drug 20mg	33
Placebo	33
Drug20mg	17
Drug 20mg	12
Drug 10mg	9
Placebo	8
DRUG 20MG	8
Placebo	8
PLACEBO	7
DRUG 10MG	6
Drug10mg	5
Drugl0mg	4
Drug20mg	3

Name:**count.,.** dtype:int64

```
clean_df["ARM"]=(
    clean_df["ARM"]
    .astype("string")
    .str.strip()
    .str.upper()


clean_df["AIRM"]= clean_df["ARM"].replace({ "DRUG
    10MG": "Drug 10mg",
    "DRUG10MG":"Drug 10mg",
    "DRUG20MG":"Drug20mg",
    "DRUG20MG":"Drug 20mg",
    "PLACEBO"::"Placebo"
})

print("ARM aftercleaning:")
print(clean_df["AIRM"].value_counts(dropna=False))  URK23CS,7079
```

ARM aftercleaning: ARM

Drug20mg	73
Drug10mg	64
Placebo	56

Name:count, dtype:Int64

```python
print("SEX after cleaning:")
print(clean_df["SEX"].value_counts(dropna=False))

print("\nARM after cleaning:")
print(clean_df["ARM"].value_counts(dropna=False))
```

```
SEX after cleaning:
SEX
F    96
M    84
Name: count, dtype: Int64

ARM after cleaning:
ARM
Drug 20mg    68
Drug 10mg    60
Placebo      52
Name: count, dtype: Int64
```

6. Result / Conclusion

The structural aim of Experiment No. 2 has been completely achieved. The clinical data cleaning program successfully compressed the highly fragmented 193-row raw file matrix down into an audited, standardized, and integrated 180-row analysis-ready dataset. Imputation of numerical arrays using features' historical local medians eliminated data gaps across all variables without introducing distribution skewness. Furthermore, removing multiple exact duplicate records and matching categorical fields (such as SEX and ARM) resolved system inconsistencies. The resulting tidy, uniform output is now fully optimized for clinical trial statistical inferences and exploratory modeling.

Student Signature	Faculty Signature	Date