

1.demographicsfeautres

December 5, 2025

1 Demographics Feature Script

1.1 Description

This script extracts baseline patient characteristics and admission metadata from `admissions.csv` and `patients.csv`.

1.2 Clinical Justification for HAPI Research

Intrinsic patient factors significantly alter tissue tolerance and HAPI susceptibility:

- * **Age:** Older adults often exhibit decreased skin elasticity, reduced subcutaneous fat, and poorer capillary perfusion.
- * **Gender:** Included to control for physiological differences in body mass distribution and skin structure.
- * **Admission Type/Timing:** “Weekend” or “Emergency” admissions may correlate with variations in staffing levels or delays in initial skin assessments.

1.3 Inputs & Outputs

- **Inputs:** `admissions.csv`, `patients.csv`
- **Output:** `demographics_feat.csv`
- **Key Features:**
 - `age`: Patient age at admission.
 - `male_flag`: Binary gender indicator.
 - `elective_adm_flag`: Controls for planned vs. unplanned care.
 - `weekend_admit_flag`: Proxy for staffing/resource variation.

```
[1]: # Import Libraries
import pandas as pd
import os
```

```
[2]: # Configure Paths
BASE_DIR = r"D:\School\5141"

# Paths to source tables
ADMISSIONS_PATH = os.path.join(BASE_DIR, "admissions.csv", "admissions.csv")
PATIENTS_PATH    = os.path.join(BASE_DIR, "patients.csv", "patients.csv")

# Output path for engineered demographic features
OUTPUT_PATH      = os.path.join(BASE_DIR, "demographics_feat.csv")
```

```
[3]: # Load Admissions & Patient Data
def load_data():
    """
    Load admissions and patient tables.
    Keep only the columns needed for demographic features to save memory.
    """

    # Admissions table: hospital stays and admission data
    adm = pd.read_csv(
        ADMISSIONS_PATH,
        usecols=["hadm_id", "subject_id", "admittime", "admission_type"],
        low_memory=False
    )

    # Patients table
    pat = pd.read_csv(
        PATIENTS_PATH,
        usecols=["subject_id", "anchor_age", "gender"],
        low_memory=False
    )

    # Ensure consistent ID types for merging
    adm["hadm_id"] = adm["hadm_id"].astype("Int64")
    adm["subject_id"] = adm["subject_id"].astype("Int64")
    pat["subject_id"] = pat["subject_id"].astype("Int64")

    # Join patient info
    df = adm.merge(pat, on="subject_id", how="left")
    return df
```

```
[4]: # Create Demographic Features Function
def process_data(df):
    """
    Create demographic and admission-context features:
    - age
    - male_flag
    - elective_adm_flag
    - weekend_admit_flag
    """

    # Parse admission time as datetime
    df["admittime"] = pd.to_datetime(df["admittime"], errors="coerce")

    # Age: from anchor_age
    df["age"] = df["anchor_age"]

    # Gender flag: 1 = male, 0 = not male
```

```

df["male_flag"] = (df["gender"].str.upper() == "M").astype(int)

# Elective admission flag: 1 if admission_type contains "elective"
df["elective_adm_flag"] = (
    df["admission_type"].str.lower().str.contains("elective", na=False)
).astype(int)

# Weekend admission flag: 1 if admitted on Saturday (5) or Sunday (6)
df["weekend_admit_flag"] = df["admittime"].dt.dayofweek.isin([5, 6]).  

    astype(int)

# Keep one row per hadm_id with the feature columns
feat = df[[
    "hadm_id",
    "age",
    "male_flag",
    "elective_adm_flag",
    "weekend_admit_flag"
]].drop_duplicates(subset=["hadm_id"])

return feat

# Save Processed Features
def save_data(df):
    """Save the final demographics feature table to CSV."""
    df.to_csv(OUTPUT_PATH, index=False)
    print(f"Saved {len(df)} rows to {OUTPUT_PATH}")

```

```
[5]: # Execute
if __name__ == "__main__":
    df = load_data()
    feat = process_data(df)
    save_data(feat)
```

Saved 546028 rows to D:\School\5141\demographics_feat.csv