US Accidents Dataset: Cleaning Report

Team Members: Sawab Hussein, Bejar Zawity, Nor Jameel,
Mohammed Salah, Nechirvan Hassan

1 Introduction

This report documents the process of exploring and cleaning the US Accidents dataset, which initially contained over 7.7 million rows and 46 columns. The dataset was large and memory-intensive, requiring careful handling to ensure reproducibility and efficiency. The final cleaned dataset is stored at:

`data/processed`

2 Data Exploration

The exploration phase focused on schema auditing, memory profiling, missingness analysis, range checks, and categorical/numeric profiling.

2.1 Schema and Memory Audit

We first loaded the dataset and inspected its schema:

```
import pandas as pd
df = pd.read_csv(DATA_PATH, low_memory=False)
print(df.shape)
df.info(memory_usage="deep")
```

Findings:

- Shape: $(7,728,394 \times 46)$.
- Memory usage: $\sim$10 GB.
- High-cardinality text columns (e.g., `Description`, `Street`, `City`) consumed the most memory.

2.2 Missingness Analysis

We computed missingness per column:

```
missing_summary = df.isna().mean().sort_values(ascending=False)
print(missing_summary.head(15))
```

Key results:

- `End_Lat`, `End_Lng`: $\sim$44% missing.

- `Precipitation(in)`: 28.5% missing.
- `Wind_Chill(F)`: 25.8% missing.
- Weather-related columns had 1–3% missingness.

2.3 Range and Sanity Checks

We validated plausible ranges:

```
# Example checks
(df['Severity'].between(1,4)).all()
(df['Distance(mi)'] >= 0).all()
(df['Start_Lat'].between(-90,90)).all()
(df['Start_Lng'].between(-180,180)).all()
```

Findings:

- Severity values were valid (1–4).
- Distances were non-negative, max ∼442 miles.
- Some negative temperatures (likely sensor errors).
- End coordinates often missing or out-of-bounds.

2.4 Temporal and Spatial Validation

- Monthly counts showed steady growth from 2016–2022, with anomalies in 2020–2021 (pandemic effects).
- Duration statistics revealed extreme outliers (>7 days).
- Duplicate coordinates were common (over 4.8M).

3 Data Cleaning

The cleaning pipeline was implemented in JupyterLab. Below are the key steps, code, and justifications.

3.1 Setup and Chunked Loading

```
import pandas as pd
chunk_size = 100_000
chunks = pd.read_csv(file_path, chunksize=chunk_size, low_memory=False)
df = pd.concat(chunks, ignore_index=True)
```

Justification: Chunked loading prevented memory crashes with the 7.7M-row dataset.

3.2 Datetime Conversion

```
for col in ["Start_Time", "End_Time", "Weather_Timestamp"]:
    df[col] = pd.to_datetime(df[col], errors="coerce")
```

Justification: Ensured temporal consistency and enabled duration calculations.

3.3 Duplicate Removal

```
df = df.drop_duplicates()
```

Justification: Removed redundant rows, though none were found in practice.

3.4 Missing Value Treatment

```
# Numerical columns: fill with mean
num_cols_mean = ["Temperature(F)", "Humidity(%)", "Pressure(in)", "
    Visibility(mi)"]
for col in num_cols_mean:
    df[col] = df[col].fillna(df[col].mean())

# Categorical columns: fill with mode
cat_cols_mode = ["Street", "City", "Zipcode", "Timezone", "Airport_Code
    ", "Wind_Direction"]
for col in cat_cols_mode:
    df[col] = df[col].fillna(df[col].mode()[0])

# Twilight-related: fill with mode
twilight_cols = ["Sunrise_Sunset", "Civil_Twilight", "Nautical_Twilight
    ", "Astronomical_Twilight"]
for col in twilight_cols:
    df[col] = df[col].fillna(df[col].mode()[0])

# Description: placeholder
df["Description"] = df["Description"].fillna("Unknown")
```

Justification: Mean imputation for continuous variables preserves distribution; mode imputation for categorical ensures consistency.

3.5 Dropping High-Missingness Columns

```
drop_cols = ["End_Lat", "End_Lng", "Wind_Chill(F)", "Precipitation(in)"
    ]
df = df.drop(columns=drop_cols)
```

Justification: Columns with >25% missingness were dropped to avoid bias and unreliable imputations.

3.6 Range and Sanity Filters

```
df = df[(df["Temperature(F)"] > -65) & (df["Temperature(F)"] < 135)]
df = df[df["Visibility(mi)"] >= 0]
df = df[df["Wind_Speed(mph)"] >= 0]
df = df[df["Pressure(in)"] > 0]
df = df[df["Distance(mi)"] >= 0]
```

Justification: Removed implausible sensor readings and ensured physical validity.

3.7 Feature Engineering

```
df["Duration_sec"] = (df["End_Time"] - df["Start_Time"]).dt.
    total_seconds()
df = df[df["Duration_sec"] <= 7*24*3600]  # remove >7 days
df["Year"] = df["Start_Time"].dt.year
df["Month"] = df["Start_Time"].dt.month
df["Hour"] = df["Start_Time"].dt.hour
df["DayOfWeek"] = df["Start_Time"].dt.day_name()
```

Justification: Derived features (duration, year, month, hour, weekday) enable temporal analysis and modeling.

3.8 Final Checks and Saving

```
print("Final␣shape:", df.shape)
print("Remaining␣missing␣values:", df.isna().sum().sum())
df.to_csv("data/processed/cleaned_dataset.csv", index=False)
```

Result: Final dataset had ∼6.98M rows and 47 columns, with only ∼106k missing values remaining.

4 Conclusion

Through systematic exploration and cleaning, we reduced noise, handled missingness, and engineered useful features. The cleaned dataset is now ready for downstream modeling and analysis. The reproducible pipeline ensures scalability and transparency.