WEEK 2

Data Cleaning I

Production-ready data cleaning skills for AI and ML pipelines

Day 6	**Missing Values**	Detection, quantification, basic handling strategies
Day 7	**Imputation Strategies**	Column-specific rules, indicators, train/test leakage
Day 8	**Duplicate Records**	Entity vs event, deduplication, aggregation
Day 9	**Data Types**	Numeric, datetime, mixed-type, schema stability
Day 10	**Outliers & Ethics**	IQR, z-score, capping, transformation, fairness

Tooling

pandas for table operations, numpy for numerical work

isna, dropna, fillna, duplicated, drop_duplicates, to_numeric, to_datetime, clip, np.log1p

```
import pandas as pd
import numpy as np
np.random.seed(42)
```

DAY 10

Outliers & Ethics

Detection, Handling, and Fairness Implications

OBJECTIVES

- Define statistical vs domain outliers
- Apply IQR and z-score methods
- Use clip and np.log1p to mitigate extremes
- Reason about fairness risks of outlier handling
- Understand model sensitivity to outliers

ACTIVITY

Compare models under different outlier strategies on heavy-tailed data.

ASSESSMENT

Explain when outlier should be removed vs transformed.

Describe capping fairness risks.

What Outliers Represent

Statistical vs domain outliers - not automatically bad data

Statistical Outliers

Values far from bulk of distribution (beyond 1.5×IQR)

May be errors OR important signals about rare events

Domain Outliers

- Impossible under domain knowledge (negative age)
- Rare but valid extremes (very high incomes)

Context determines if outlier is valid

EXAMPLE WITH ONE EXTREME VALUE

```
df_out = pd.DataFrame({"user_id": range(1, 8),
    "income": [40000, 45000, 50000, 52000, 55000, 60000, 1000000]})
print("Mean income:", df_out["income"].mean())  # Heavily skewed by outlier!
```

DAY 10 · OUTLIERS

IQR Method for Outlier Detection

Robust to moderate skew using quantiles

Bounds Formula

lower = Q1 − k × IQR

upper = Q3 + k × IQR

Typical k: 1.5 (moderate), 3 (extreme)

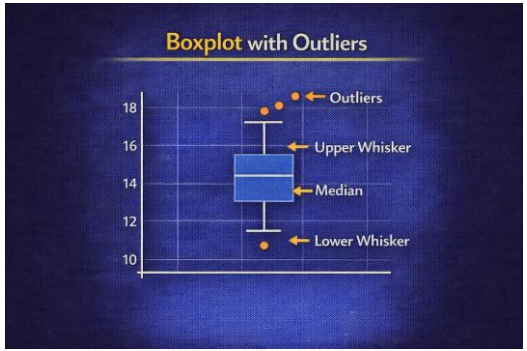
Why IQR

- Based on quartiles, not mean/std
- Less affected by extreme values themselves
- Standard method for box plots

IQR OUTLIER DETECTION

```
def iqr_bounds(s, k=1.5):
    q1, q3 = s.quantile(0.25), s.quantile(0.75)
    iqr = q3 - q1
    return q1 - k * iqr, q3 + k * iqr

lower, upper = iqr_bounds(df_out["income"], k=1.5)
mask_iqr = (df_out["income"] < lower) | (df_out["income"] > upper)
```



DAY 10 · OUTLIERS

Z-Score Method for Outlier Detection

Standard deviations from mean - assumes normality

Formula

z = (x − mean) / std

Rule of thumb: |z| > 3 often used as outlier cut

Caveats

- Assumes approximate normality
- Outliers themselves affect mean/std!
- May miss outliers in heavy-tailed data
- Better for symmetric distributions

Z-SCORE OUTLIER DETECTION

```
income = df_out["income"].astype(float)
mean, std = income.mean(), income.std(ddof=0)

df_out["income_z"] = (income - mean) / std
mask_z = df_out["income_z"].abs() > 3
print("Z-score outliers:\n", df_out[mask_z])
```

DAY 10 · OUTLIERS

Handling Outliers — Removal, Capping, Transformation

Each strategy has statistical and ethical implications

Removal

Use when outliers are clear data errors

Risk: Losing valid extreme cases

Capping (clip)

Replace below/above bounds with bounds

Preserves row, limits influence

Transform (log1p)

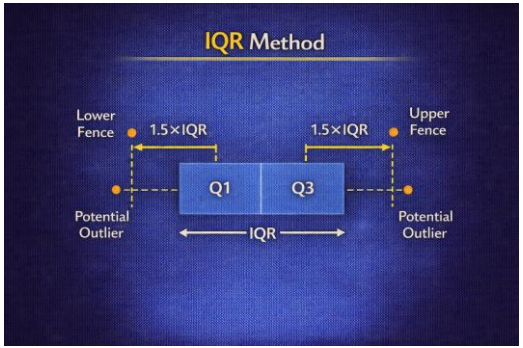
Compress heavy tails

Keeps zeros valid, reduces skew

THREE STRATEGIES

```
lower, upper = iqr_bounds(df_out["income"], k=1.5)

df_no_out = df_out[~((df_out["income"] < lower) | (df_out["income"] > upper))]  # Remove
df_capped = df_out.copy()
df_capped["income_capped"] = df_capped["income"].clip(lower=lower, upper=upper)  # Cap
df_trans = df_out.copy()
df_trans["income_log1p"] = np.log1p(df_trans["income"])  # Transform
```



DAY 10 · OUTLIERS

Model Sensitivity & Ethical Considerations

Outlier handling changes who models focus on

Model Sensitivity

- Logistic/linear regression dominated by extreme points
- After capping or log1p, coefficients and decision boundaries change
- Tree models more robust but still affected

Ethical Concerns

- Removing rare high-risk cases (e.g., very high medical costs) hides important signals
- If outlier regions correspond to specific groups → disparate impact

COEFFICIENT COMPARISON

```
from sklearn.linear_model import LogisticRegression
coef_raw = LogisticRegression().fit(X_raw, y).coef_[0][0]
coef_cap = LogisticRegression().fit(X_cap, y).coef_[0][0]
print("Coef raw:", coef_raw, "Coef capped:", coef_cap)  # Different!
```

DAY 10 · ACTIVITY & ASSESSMENT

Outliers Practice

Compare strategies on heavy-tailed data

Activity

1. Use dataset with heavy-tailed numerics (income, transaction)
2. Apply IQR and z-score detection, compare sets
3. Build three model variants:
   - No outlier handling
   - IQR capping with clip
   - log1p transformation
4. Compare performance by feature quantile/group

Assessment

Explain when an outlier should be removed vs transformed in a lending model.

Describe a scenario where capping could disadvantage a specific group and how you would detect that risk.

SKELETON

```
def detect_outliers_iqr(df, col, k=1.5):
    lower, upper = iqr_bounds(df[col], k=k)
    return (df[col] < lower) | (df[col] > upper)
```

Week 2 Complete!

Data Cleaning I — Key Patterns Checklist

Missing Values	Detect, normalize tokens, quantify; decide drop vs impute vs indicator
Imputation	Column-specific strategies, indicators where informative, fit on train only
Duplicates	Define entity/event uniqueness, sort, deduplicate, aggregate to stable features
Data Types	Normalize with to_numeric, to_datetime (errors='coerce'), ensure schema stability
Outliers	IQR or z-score detection; removal, clip, or log1p with domain and ethics in mind

Ethics & ML Reminder
For each cleaning decision, ask which users or groups are most affected. Inspect missingness, duplicates, and outliers by subgroup, not only globally.

END-TO-END CLEANING SKETCH

```
def basic_clean(df):
    df = df.replace(["N/A", "not reported"], np.nan)
    df["income_num"] = pd.to_numeric(df["income"], errors="coerce").fillna(df["income"].median())
    return df.clip(lower=iqr_bounds(df["income_num"])[0], upper=iqr_bounds(df["income_num"])[1])
```