data_cleaning_lab2_simple

July 16, 2025

1 Data Cleaning Exercise - Laboratory 2

AICTE Faculty ID: 1-3241967546
Faculty Name: Milav Jayeshkumar Dabgar
Date: July 16, 2025

1.1 Objective

Perform data cleaning and preprocessing on the Car Evaluation dataset.

1.2 1. Import Libraries

[1]:
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

print("Libraries imported!")
```

```
Libraries imported!
```

1.3 2. Load Dataset

[2]:
```
# Load car evaluation dataset
data = pd.read_csv('/Users/milav/Code/qip-dl/ml/day-2/car_evaluation.csv',␣
 ↪header=None)

print(f"Dataset shape: {data.shape}")
print("\nFirst few rows:")
print(data.head())
```

```
Dataset shape: (1728, 7)

First few rows:
       0      1  2  3      4     5      6
0  vhigh  vhigh  2  2  small   low  unacc
1  vhigh  vhigh  2  2  small   med  unacc
2  vhigh  vhigh  2  2  small  high  unacc
```

```
3  vhigh  vhigh  2  2    med   low  unacc
4  vhigh  vhigh  2  2    med   med  unacc
```

1.4 3. Data Analysis

[3]:
```
# Add column names
columns = ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']
data.columns = columns

print("Dataset with column names:")
print(data.head())

print(f"\nRows: {data.shape[0]}, Columns: {data.shape[1]}")
```

```
Dataset with column names:
  buying  maint doors persons lug_boot safety  class
0  vhigh  vhigh     2       2    small    low  unacc
1  vhigh  vhigh     2       2    small    med  unacc
2  vhigh  vhigh     2       2    small   high  unacc
3  vhigh  vhigh     2       2      med    low  unacc
4  vhigh  vhigh     2       2      med    med  unacc

Rows: 1728, Columns: 7
```

[4]:
```
# Check categorical values in each column
for col in data.columns:
    print(f"\n{col}: {data[col].unique()}")
```

```
buying: ['vhigh' 'high' 'med' 'low']

maint: ['vhigh' 'high' 'med' 'low']

doors: ['2' '3' '4' '5more']

persons: ['2' '4' 'more']

lug_boot: ['small' 'med' 'big']

safety: ['low' 'med' 'high']

class: ['unacc' 'acc' 'vgood' 'good']
```

1.5 4. Check Missing Values

[5]:
```
# Check for missing values
print("Missing values:")
print(data.isnull().sum())
```

```
if data.isnull().sum().sum() == 0:
    print("\nNo missing values found!")
```

```
Missing values:
buying      0
maint       0
doors       0
persons     0
lug_boot    0
safety      0
class       0
dtype: int64

No missing values found!
```

1.6 5. Feature Engineering

[6]:
```
# Convert doors and persons to numeric
processed_data = data.copy()

# Convert doors
door_mapping = {'2': 2, '3': 3, '4': 4, '5more': 5}
processed_data['doors'] = processed_data['doors'].map(door_mapping)

# Convert persons
person_mapping = {'2': 2, '4': 4, 'more': 6}
processed_data['persons'] = processed_data['persons'].map(person_mapping)

print("After converting doors and persons:")
print(processed_data.dtypes)
```

```
After converting doors and persons:
buying      object
maint       object
doors        int64
persons      int64
lug_boot    object
safety      object
class       object
dtype: object
```

[7]:
```
# Encode categorical variables
categorical_cols = ['buying', 'maint', 'lug_boot', 'safety', 'class']

for col in categorical_cols:
    le = LabelEncoder()
    processed_data[col] = le.fit_transform(processed_data[col])
```

```
    print(f"{col} encoded: {processed_data[col].unique()}")

print("\nEncoded dataset:")
print(processed_data.head())
```

```
buying encoded: [3 0 2 1]
maint encoded: [3 0 2 1]
lug_boot encoded: [2 1 0]
safety encoded: [1 2 0]
class encoded: [2 0 3 1]

Encoded dataset:
   buying  maint  doors  persons  lug_boot  safety  class
0       3      3      2        2         2       1      2
1       3      3      2        2         2       2      2
2       3      3      2        2         2       0      2
3       3      3      2        2         1       1      2
4       3      3      2        2         1       2      2
```

1.7 6. Separate Features and Target

[8]:
```
# Separate independent and dependent variables
X = processed_data.drop('class', axis=1)
y = processed_data['class']

print(f"Features shape: {X.shape}")
print(f"Target shape: {y.shape}")

print("\nFeature columns:", list(X.columns))
print("Target classes:", y.unique())
```

```
Features shape: (1728, 6)
Target shape: (1728,)

Feature columns: ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety']
Target classes: [2 0 3 1]
```

1.8 7. Train-Test Split

[9]:
```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print(f"Training set: {X_train.shape}")
print(f"Testing set: {X_test.shape}")

print("\nData preprocessing completed!")
```

```
print("Dataset is ready for machine learning.")
```

```
Training set: (1382, 6)
Testing set: (346, 6)

Data preprocessing completed!
Dataset is ready for machine learning.
```

1.9 8. Summary

1.9.1 Data Cleaning Results:

Successfully processed the **Car Evaluation dataset** with the following characteristics:

Dataset Profile: - **Size**: 1,728 samples × 7 features - **Data Quality**: No missing values detected - **Feature Types**: All categorical features requiring encoding

Key Transformations Applied:

1. **Column Naming**: Added meaningful names: `buying`, `maint`, `doors`, `persons`, `lug_boot`, `safety`, `class`
2. **Categorical Analysis**: Identified distinct categories:
   - `buying/maint`: 4 levels (vhigh, high, med, low)
   - `doors`: 4 levels (2, 3, 4, 5more)
   - `persons`: 3 levels (2, 4, more)
   - `lug_boot`: 3 levels (small, med, big)
   - `safety`: 3 levels (low, med, high)
   - `class`: 4 levels (unacc, acc, vgood, good)
3. **Numerical Conversion**:
   - `doors`: Mapped to 2, 3, 4, 5
   - `persons`: Mapped to 2, 4, 6
4. **Label Encoding**: Converted categorical variables to numerical format for ML compatibility
5. **Data Splitting**:
   - Training: 1,382 samples (80%)
   - Testing: 346 samples (20%)
   - Stratified split maintains class distribution

1.9.2 Final Dataset Status:

- **Features**: 6 numerical columns (buying, maint, doors, persons, lug_boot, safety)
- **Target**: 1 categorical column (class) with 4 classes
- **Ready for ML**: All preprocessing steps completed successfully

1.9.3 Data Cleaning Pipeline Validated:

Load → Analyze → Clean → Transform → Split

The dataset is now properly formatted and ready for machine learning model training and evaluation.