Welcome to the CoGrammar Data Cleaning

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

CoGrammar

Data Science Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Data Science Session Housekeeping cont.

- For all **non-academic questions**, please submit a query:
  www.hyperiondev.com/support

- Report a **safeguarding** incident:
  www.hyperiondev.com/safeguardreporting

- We would love your **feedback** on lectures: **Feedback on Lectures**

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

Data Cleaning

April 2024

Data Cleaning

- Data cleaning is a crucial step in the data science pipeline
- Ensures **data quality and reliability** for analysis and modeling
- Common data quality issues include **missing data, duplicates, inconsistent formatting, and outliers**

CoGrammar

Learning objectives

- ❖ Describe techniques for **handling missing data** and when each is appropriate to use.
- ❖ Demonstrate how to identify and remove duplicate records in a dataset using Pandas.
- ❖ Explain the importance of **consistent data formatting and apply methods to standardise data.**
- ❖ Define **outliers** and discuss strategies for detecting and handling them appropriately based on the data context.

CoGrammar

Missing Data

CoGrammar

Handling Missing Data

- Missing data refers to the **absence of values in one or more variables in a dataset.**

- Identifying missing values:
  - Look for **null, NaN, or empty cells in the dataset**.
  - Use functions like **isnull()** or **isna()** in Pandas

```
# Identify missing values
df_missing.isnull().sum()
```

✓ 0.0s

```
total_bill    10
tip            0
sex           10
smoker         0
day            0
time           0
size           0
dtype: int64
```

CoGrammar

Understand Missing Data Mechanisms

- These mechanisms are more important if you do research in the field, so we're going to glaze over it
- It helps us understand what techniques to use, but **we could intuit it in most cases**
- Here is the [original paper]

CoGrammar

Understand Missing Data Mechanisms

- **MCAR:** Missing Completely at Random (missingness unrelated to any variables)
  - Smoking status is not recorded in a random sample of patients
- **MAR:** Missing at Random (missingness depends on observed variables)
  - Smoking status is not documented in female patients because the doctor was to shy to ask 🙂
- **MNAR:** Missing Not at Random (missingness depends on missing values themselves)
  - Smoking status is not recorded in patients admitted as an emergency, who are also more likely to have worse outcomes from surgery

Techniques for Handling Missing Data

- **Deletion:** Remove records with missing values (only suitable if missing data is minimal and random).
  - Suitable for random missingness
  - Not the first resort, dropping data means losing some important context or skewing the dataset in some cases

```python
df.shape
```

✓ 0.0s

```
(244, 7)
```

```python
# Deletion: Remove records with missing values
df_deleted = df_missing.dropna()
df_deleted.shape
```

✓ 0.0s

```
(225, 7)
```

CoGrammar

Techniques for Handling Missing Data

❖ **Imputation:** Fill in missing values with estimated or calculated values.

➢ Simple imputation: Mean, median, or mode imputation

```
# Simple Imputation: Fill missing values with mean for numeric columns and mode for categorical
# columns
df_imputed = df_missing.copy()
df_imputed['total_bill'] = df_imputed['total_bill'].fillna(df_imputed['total_bill'].mean())
df_imputed['sex'] = df_imputed['sex'].fillna(df_imputed['sex'].mode()[0])
```

➢ Advanced imputation: K-Nearest Neighbors (KNN), Multiple Imputation by Chained Equations (MICE)

■ We'll get to KNN in another lecture 🤩

Techniques for Handling Missing Data

```python
# Advanced Imputation: KNN Imputation
from sklearn.impute import KNNImputer

# Create a copy of the dataset for KNN imputation
df_imputed_knn = df_missing.copy()

# Initialize and fit the KNN imputer
imputer = KNNImputer(n_neighbors=5)
df_imputed_knn[['total_bill', 'tip', 'size']] = imputer.fit_transform(
    df_imputed_knn[['total_bill', 'tip', 'size']]
)
```

CoGrammar

Which of the following is NOT a common data quality issue?

A. Missing values
B. Duplicates
C. Inconsistent formatting
D. Small sample size

CoGrammar

Which of the following is NOT a common data quality issue?

A. Missing values

B. Duplicates

C. Inconsistent formatting

D. Small sample size

CoGrammar

Which technique is suitable for handling missing data only if the amount is minimal and missing at random?

A. Mean imputation

B. Deletion

C. K-Nearest Neighbours imputation

D. Multiple Imputation by Chained Equations

Which technique is suitable for handling missing data only if the amount is minimal and missing at random?

A. Mean imputation

B. Deletion

C. K-Nearest Neighbours imputation

D. Multiple Imputation by Chained Equations

CoGrammar

Duplicates

CoGrammar

Dealing with Duplicates

- Identify duplicates using functions like duplicated() in Pandas

```python
# Show all duplicated rows
df_duplicates[df_duplicates.duplicated(keep=False)]
```

keep = False just marks all duplicates.

	total_bill	tip	sex	smoker	day	time	size
46	22.23	5.00	Male	No	Sun	Dinner	2
92	5.75	1.00	Female	Yes	Fri	Dinner	2
123	15.95	2.00	Male	No	Thur	Lunch	2
158	13.39	2.61	Female	No	Sun	Dinner	2
198	13.00	2.00	Female	Yes	Thur	Lunch	2
202	13.00	2.00	Female	Yes	Thur	Lunch	2
234	15.53	3.00	Male	Yes	Sat	Dinner	2
244	22.23	5.00	Male	No	Sun	Dinner	2
245	15.53	3.00	Male	Yes	Sat	Dinner	2
246	13.39	2.61	Female	No	Sun	Dinner	2
247	5.75	1.00	Female	Yes	Fri	Dinner	2
248	15.95	2.00	Male	No	Thur	Lunch	2

CoGrammar

Dealing with Duplicates

❖ Dropping duplicates is fine and encouraged, it does not cause the data to lost necessary context

```
# Remove duplicate records
df_deduplicated = df_duplicates.drop_duplicates()
```

In Pandas, which function can be used to identify duplicate records in a dataset?

A. find_duplicates()

B. duplicated()

C. is_duplicate()

D. has_duplicates()

CoGrammar

In Pandas, which function can be used to identify duplicate records in a dataset?

A. find_duplicates()
B. duplicated()
C. is_duplicate()
D. has_duplicates()

CoGrammar

Formatting and Standardization

CoGrammar

Data Formatting and Standardization

- Consistent data formatting is essential for accurate analysis and compatibility
- Common formatting issues:
  - **Date and time formats:** Ensure consistent representation (e.g., YYYY-MM-DD, HH:MM:SS)
  - **Text case inconsistencies:** Convert text to a consistent case (lowercase or uppercase)
  - **Inconsistent value representations:** Standardize values (e.g., "Yes"/"No" vs. "Y"/"N")

CoGrammar

Data Formatting and Standardization

- Techniques for standardizing data:
  - Convert date/time columns using to_datetime()
  - Convert text case using str.lower() or str.upper()
  - Map inconsistent values to standardized representations

```python
# Standardize text case for 'sex' and 'smoker' columns
df['sex'] = df['sex'].str.upper()
df['smoker'] = df['smoker'].str.title()

df.head()
```

✓ 0.0s

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	FEMALE	No	Sun	Dinner	2
1	10.34	1.66	MALE	No	Sun	Dinner	3
2	21.01	3.50	MALE	No	Sun	Dinner	3
3	23.68	3.31	MALE	No	Sun	Dinner	2
4	24.59	3.61	FEMALE	No	Sun	Dinner	4

CoGrammar

Which of the following is a technique for standardising inconsistent text case?

A. astype()

B. to_datetime()

C. upper() or lower()

D. strip()

CoGrammar

Which of the following is a technique for standardising inconsistent text case?

A. astype()

B. to_datetime()

C. upper() or lower()

D. strip()

Outliers

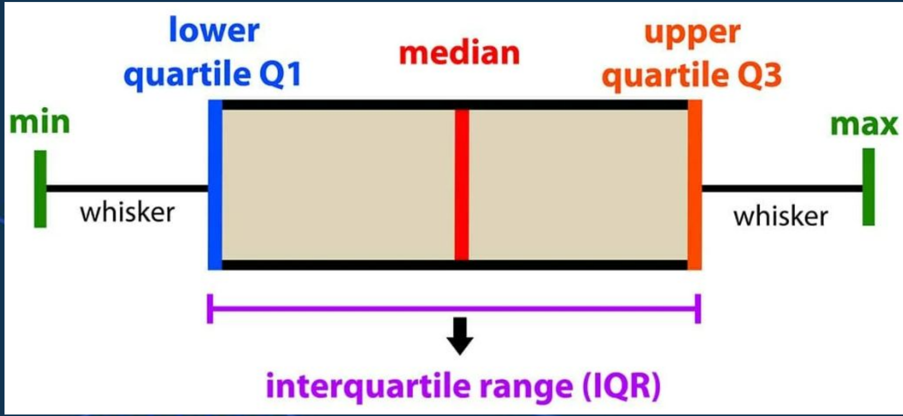
Outliers

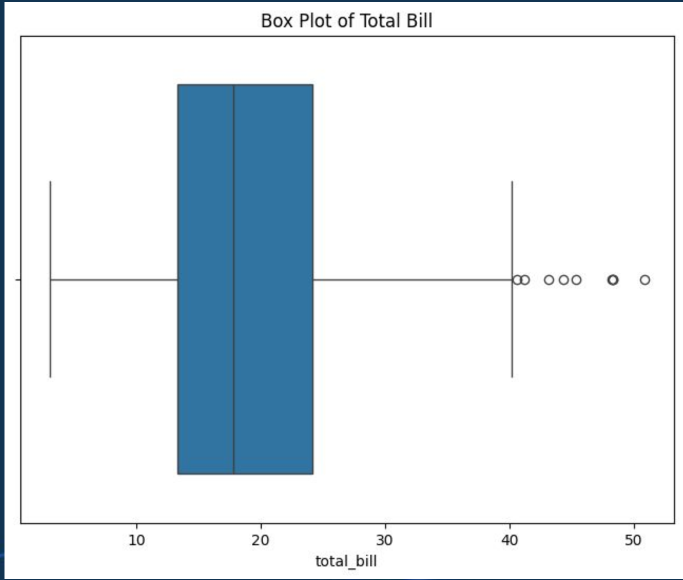
- Outliers are data points that significantly deviate from the rest of the data distribution

CoGrammar

Identifying Outliers

- Visual inspection using plots like box plots, scatter plots, or histograms





CoGrammar

Identifying Outliers

- Statistical methods like z-score or interquartile range (IQR)
  - Much less common given how good box plot already show outliers

```python
# Identify outliers using z-score
from scipy import stats

z_scores = np.abs(stats.zscore(df['total_bill']))
threshold = 2.5
outliers_zscore = np.where(z_scores > threshold)

outliers_zscore
```

✓ 0.0s

```
(array([ 59, 102, 156, 170, 182, 197, 212]),)
```

```python
# Identify outliers using Interquartile Range (IQR)
Q1 = df['total_bill'].quantile(0.25)
Q3 = df['total_bill'].quantile(0.75)
IQR = Q3 - Q1

outliers_iqr = df[(df['total_bill'] < (Q1 - 1.5 * IQR)) | (df['total_bill'] > (Q3 + 1.5 * IQR))]
len(outliers_iqr)
```

✓ 0.0s MagicPython

```
9
```

Handling Outliers

- **Removal:** Remove outliers if they are erroneous or irrelevant to the analysis
  - Use when outliers are clearly erroneous or irrelevant to the analysis
  - Be cautious, as removing outliers may result in loss of information

```
# Removal: Remove outliers
df_removed = df[~((df['total_bill'] < (Q1 - 1.5 * IQR)) | (df['total_bill'] > (Q3 + 1.5 * IQR)))]
df_removed.shape
```

✓ 0.0s MagicPython

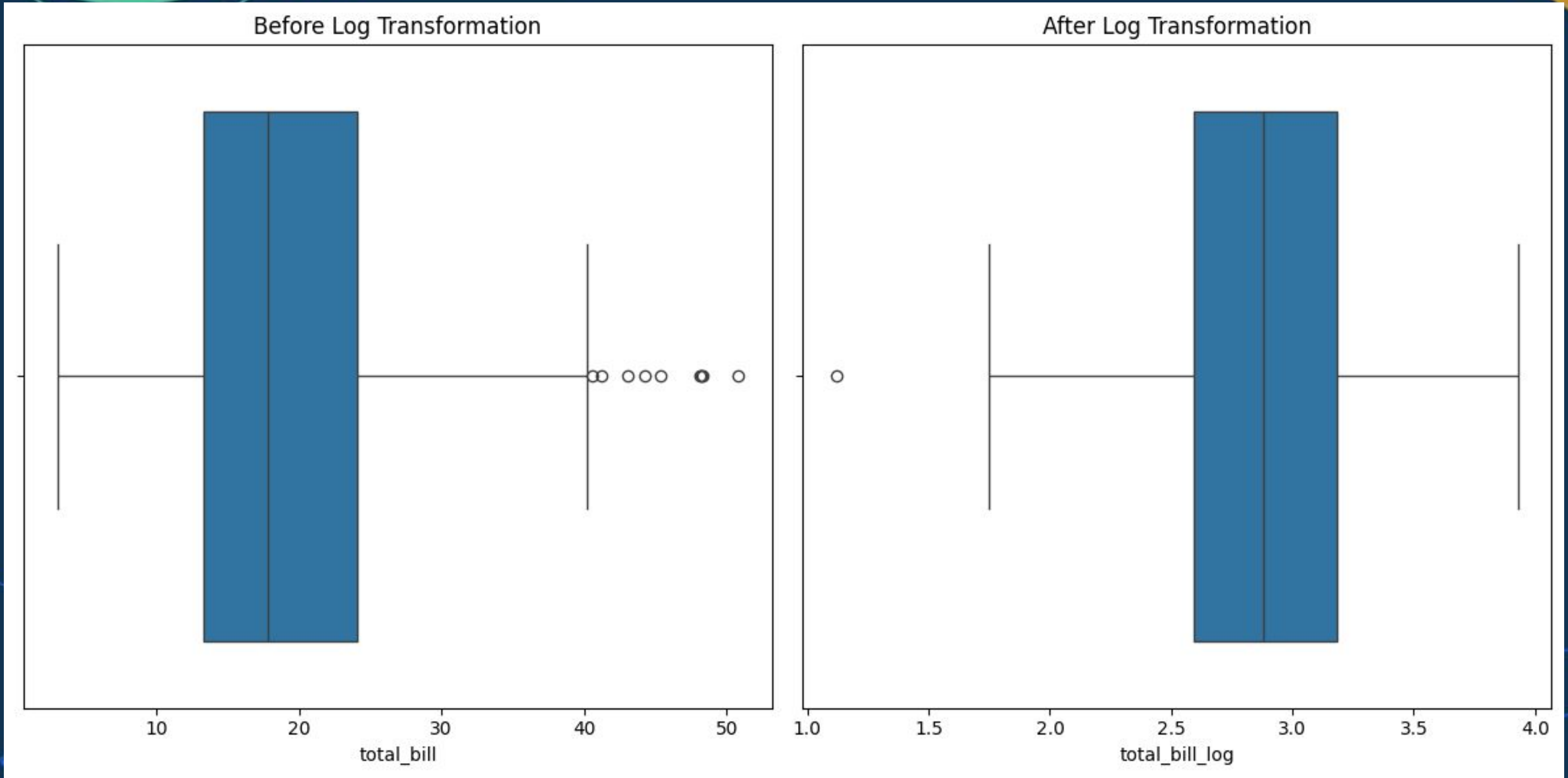
```
(235, 9)
```

Handling Outliers

- **Transformation:** Apply mathematical transformations (e.g., logarithmic, square root) to reduce the impact of outliers
  - Use when outliers are valid but have a skewed distribution
  - Helps to reduce the impact of outliers while preserving the data

CoGrammar

Handling Outliers

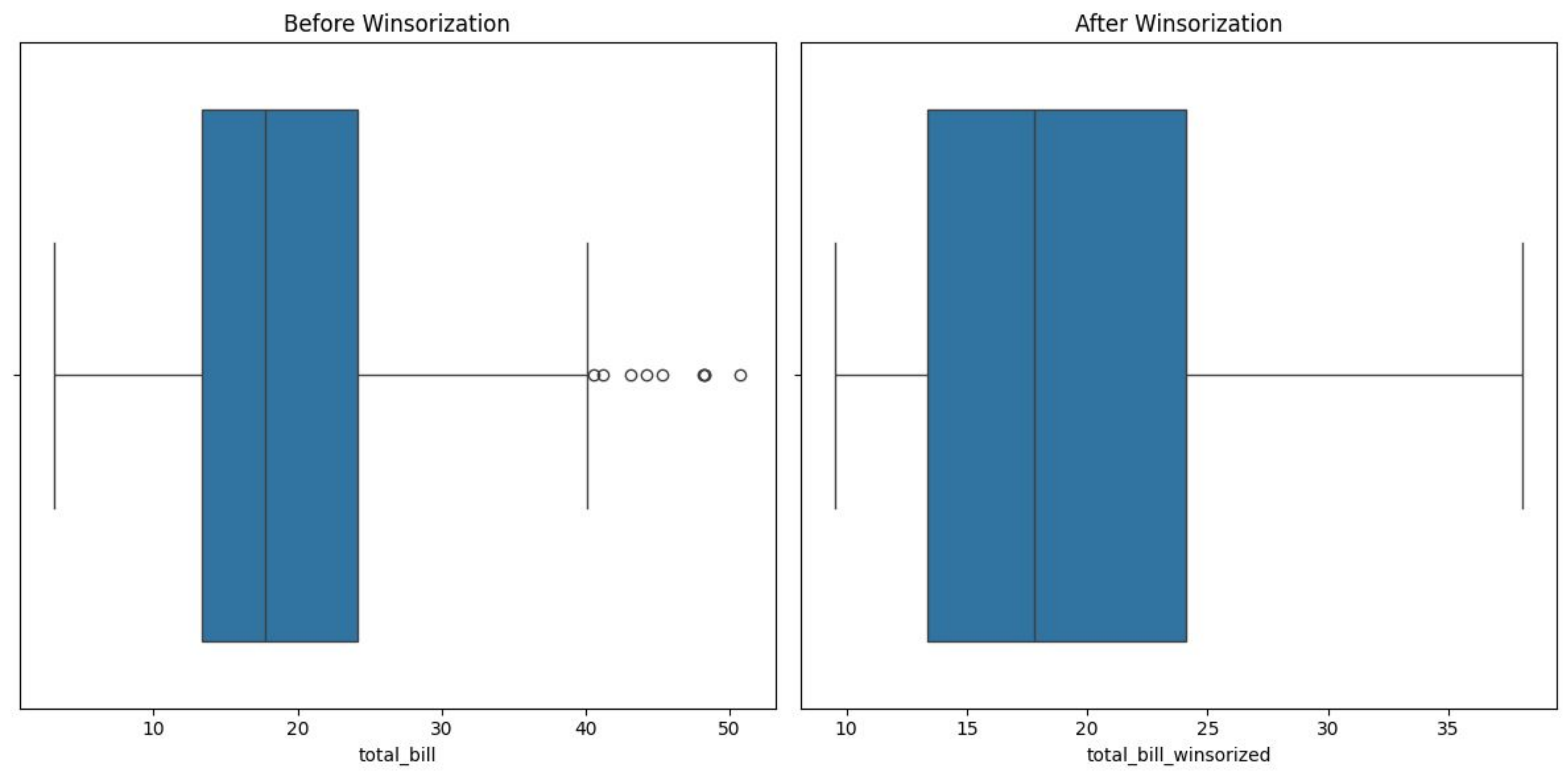


CoGrammar

Handling Outliers

- **Winsorization:** Replace extreme values with the nearest non-outlier values
  - Use when outliers are valid but need to be treated to reduce their influence
  - Maintains the overall distribution shape while limiting the extreme values

Handling Outliers



CoGrammar

Which strategy replaces outlier values with the nearest non-outlier values?

A. Removal
B. Transformation
C. Winsorisation
D. Standardisation

Which strategy replaces outlier values with the nearest non-outlier values?

A. Removal

B. Transformation

C. Winsorisation

D. Standardisation

CoGrammar

Iterative Data Cleaning

CoGrammar

Iterating

- Data cleaning is an iterative process that may require multiple rounds
- Continuously assess and refine the cleaned data based on analysis results and feedback
- Integrate data cleaning with data analysis and modeling for optimal results

Libraries

CoGrammar

fuzzywuzzy

- ❖ Provides string matching and similarity scoring functions
- ❖ Key features:
  - ➢ **Ratio:** Calculates the similarity ratio between two strings
  - ➢ **Partial Ratio:** Calculates the similarity ratio considering substrings
  - ➢ **Token Set Ratio:** Calculates the similarity ratio considering common tokens

fuzzywuzzy

```
!pip3 install fuzzywuzzy python-Levenshtein
```

```
from fuzzywuzzy import fuzz
```

```
fuzz.ratio("apple", "appel")
✓ 0.0s
80
```

```
fuzz.partial_ratio("apple", "app")
✓ 0.0s
100
```

```
# Gives a 100 if every token in the first string is in the second string
fuzz.token_set_ratio("apple orange", "orange apple")
✓ 0.0s
100
```

chardet

- Detects the encoding of a given byte string
- Key features:
  - Supports various encodings (e.g., UTF-8, ISO-8859-1, etc.)
  - Provides confidence scores for detected encodings

```
!pip3 install chardet
```

```
import chardet
```

```
byte_string = b"Hello, World!"
encoding = chardet.detect(byte_string)
encoding
```

✓ 0.0s

```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

Further Learning

KDNuggets - [Learn Data Cleaning and Preprocessing for Data Science with This Free eBook]

Kaggle - [Short Data Cleaning Course]

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar