CoGrammar

Welcome to this session:

Task Walkthrough - Tasks 13 - 15

The session will start shortly...

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

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

Skills Bootcamp Data Science

- 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]

Skills Bootcamp Data Science

- 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***

- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

Learning Outcomes

- **Clean and preprocess data** by handling whitespace, standardizing text, and formatting categorical variables.
- **Differentiate between standardization and normalization** and apply them appropriately.
- **Perform exploratory data analysis (EDA)** to detect patterns, outliers, and correlations in data.
- **Apply Linear Regression** to predict continuous numerical outcomes.
- **Apply Logistic Regression** to classify categorical outcomes.

Lecture Overview

- → Presentation of the Task
- → Data Cleaning
- → Data Preprocessing
- → EDA
- → Regression
- → Task Walkthrough

Task Walkthrough

Imagine you're a data scientist at a healthcare startup. Your team is developing a predictive model to identify patients at risk of diabetes based on their medical history. Your job? Clean, explore, and analyze the data, then build a model to make predictions!

- **Linear Regression: Predict blood sugar level based on age and BMI.**
- **Logistic Regression: Classify whether a patient is at risk of diabetes.**

By the end of this task, you'll be able to transform raw data into meaningful insights and predictive models, just like a real-world data scientist! 🚀

What does Exploratory Data Analysis (EDA) primarily help with?

A. Identifying patterns and relationships in data
B. Automating data preprocessing
C. Improving model accuracy instantly
D. Removing all outliers

Which type of regression is used for predicting a continuous numerical value?

A. Logistic Regression
B. Decision Trees
C. Linear Regression
D. K-Means Clustering

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**

HyperionDev

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
```

HyperionDev

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)
```

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 🤩

Dealing with Duplicates

- Identify duplicates using functions like duplicated() in Pandas

```
# 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

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()
```

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")

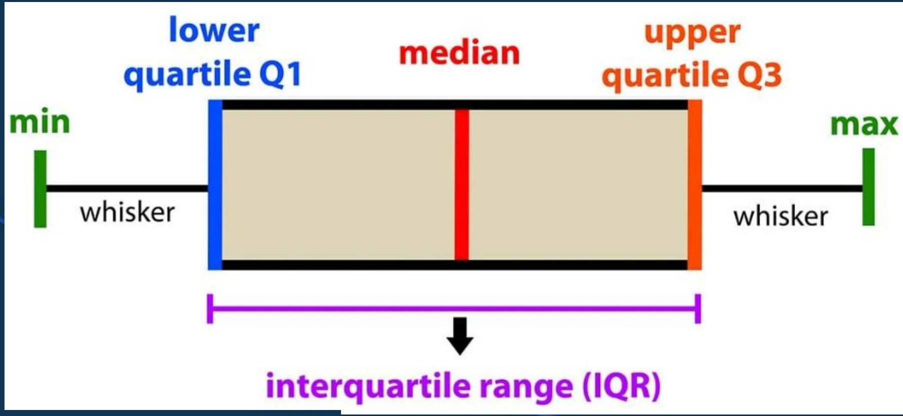
Outliers

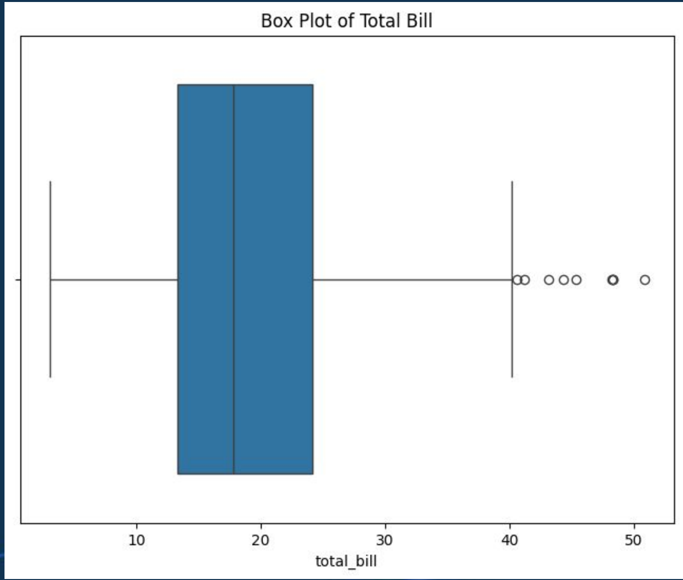
- Outliers are data points that significantly deviate from the rest of the data distribution

HyperionDev

Identifying Outliers

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





HyperionDev

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

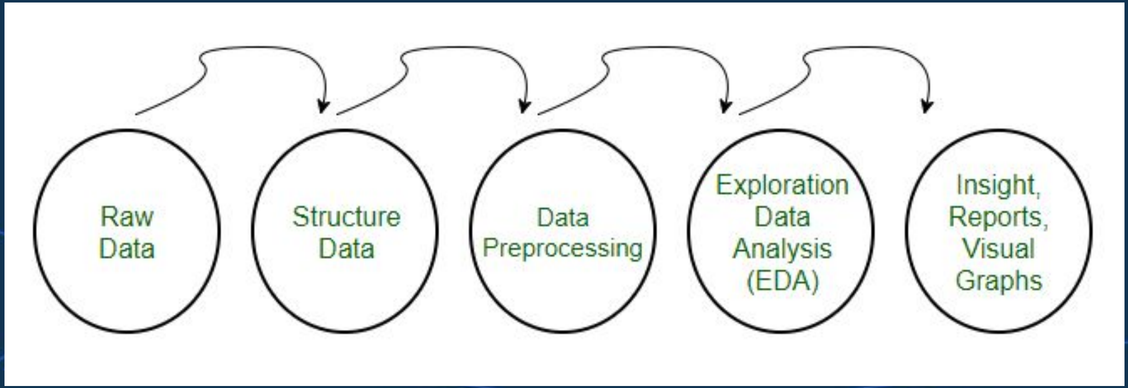
HyperionDev

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

Data Preprocessing

❖ Data preprocessing is a crucial step in the data science pipeline, going beyond basic cleaning to ensure data quality and suitability for machine learning.



Source: [GeeksForGeeks]

Feature Scaling

- Purpose: Ensure fair comparison and contribution of features
- Techniques:
  - **Standardization (Z-score normalization):** Transforms features to have zero mean and unit variance

    $$X' = \frac{X - \mu}{\sigma}$$

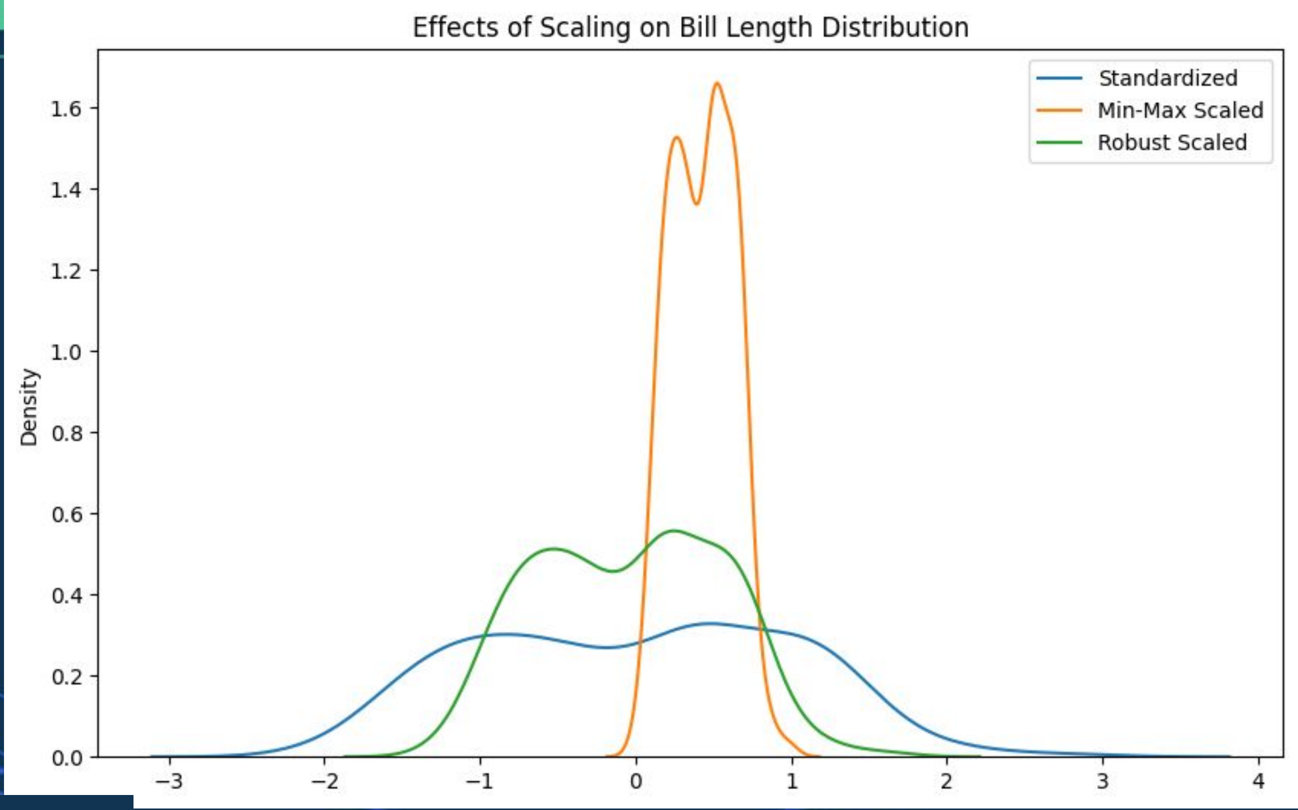
  - **Min-max scaling:** Scales features to a specific range, typically 0 to 1

    $$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

  - **Robust scaling:** Uses robust statistics (median and interquartile range) to scale features
    - (X-median)/IQR

Considerations

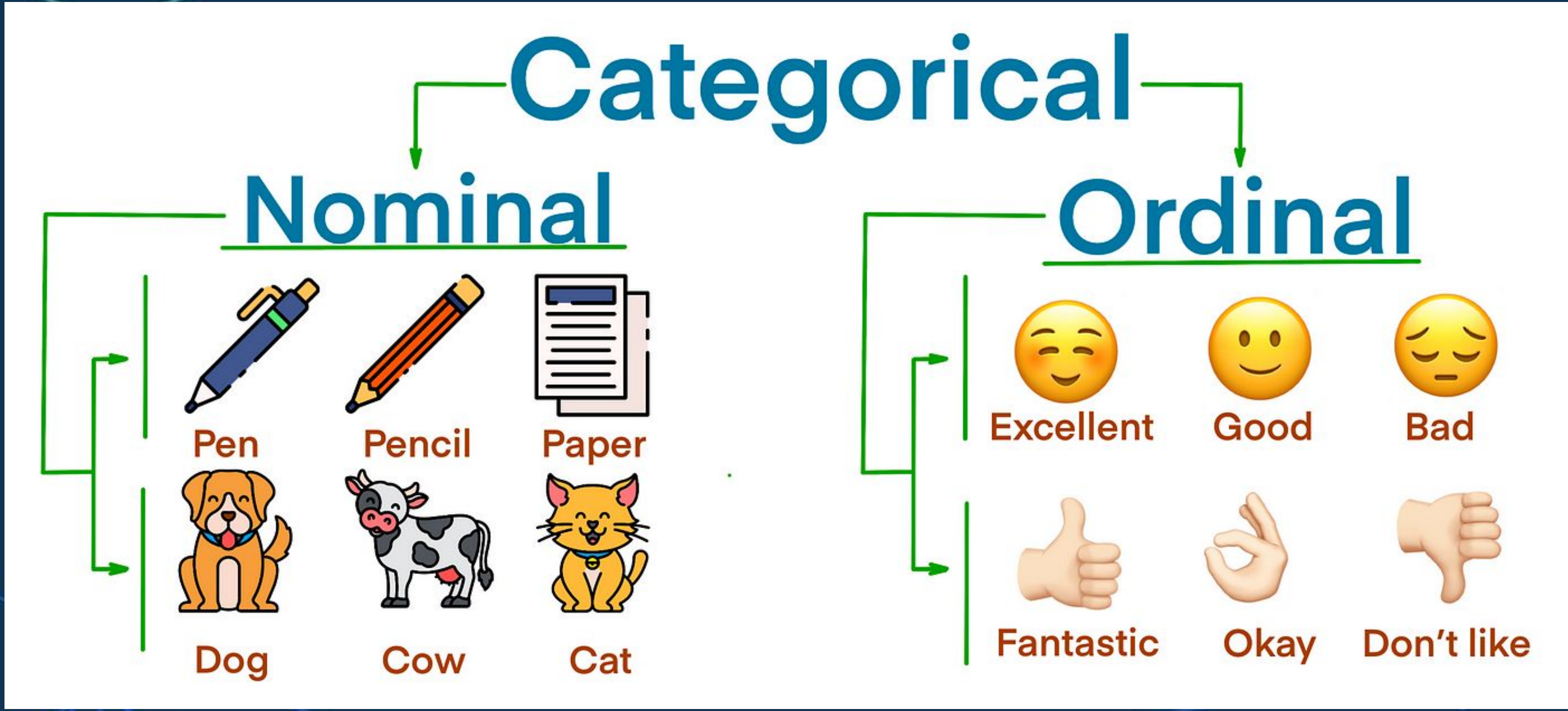
- **Standardization:** Good default, assumes Gaussian distribution
- **Min-max scaling:** Suitable for bounded features or non-Gaussian data
- **Robust scaling:** Recommended when outliers are present



HyperionDev

Nominal vs. Ordinal

- **Nominal:** Categories without inherent order (e.g., color)
- **Ordinal:** Categories with meaningful order (e.g., size)



HyperionDev

Encoding Nominal variables

- One-hot encoding: Creates binary dummy variables for each category
  - Increases dimensionality, which may impact model performance

Index	Animal
0	Dog
1	Cat
2	Sheep
3	Horse
4	Lion

One-Hot code

Index	Dog	Cat	Sheep	Lion	Horse
0	1	0	0	0	0
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	0	1
4	0	0	0	1	0

Source: [AnalyticsVidhya]

Encoding Nominal variables

- ❖ Binary encoding: Assigns unique binary codes to categories
  - ➢ Useful when the number of categories is large, and one-hot encoding leads to high dimensionality

	City
0	Delhi
1	Mumbai
2	Hyderabad
3	Chennai
4	Bangalore
5	Delhi
6	Hyderabad
7	Mumbai
8	Agra

	City_0	City_1	City_2	City_3
0	0	0	0	1
1	0	0	1	0
2	0	0	1	1
3	0	1	0	0
4	0	1	0	1
5	0	0	0	1
6	0	0	1	1
7	0	0	1	0
8	0	1	1	0

Source: AnalyticsVidhya

Encoding Ordinal variables

- Label encoding: Assigns numerical labels based on order
  - Maintains ordinal information but implies linear relationships between categories
  - May not be appropriate if the ordinal relationship is not linear

	Degree
0	1
1	4
2	2
3	3
4	3
5	4
6	5
7	1
8	1

Source: AnalyticsVidhya

Encoding Ordinal variables

❖ Ordinal encoding: Assigns numerical labels based on order

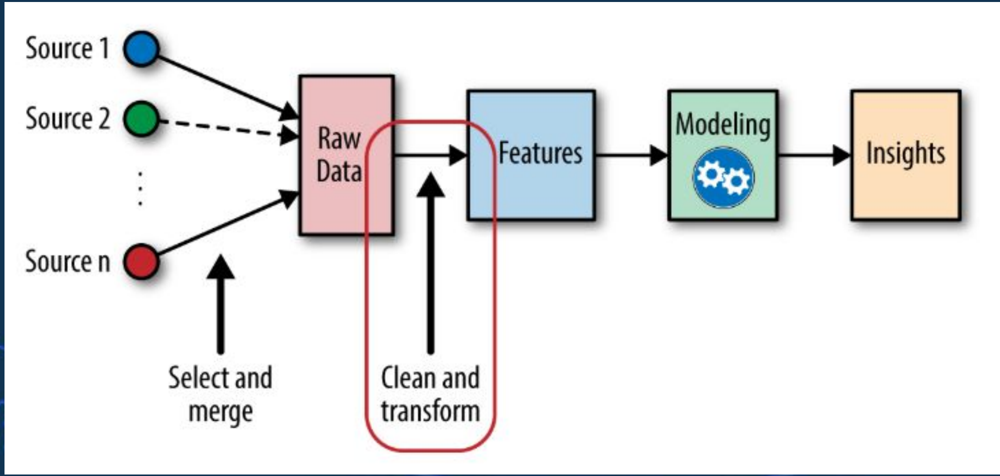
- ➢ Preserves ordinal information without implying linear relationships
- ➢ Suitable when the ordinal relationship between categories is meaningful

	Degree
0	1
1	4
2	2
3	3
4	3
5	4
6	5
7	1
8	1

Source: [AnalyticsVidhya]

Feature Engineering

❖ Create informative features that improve model performance and interpretability



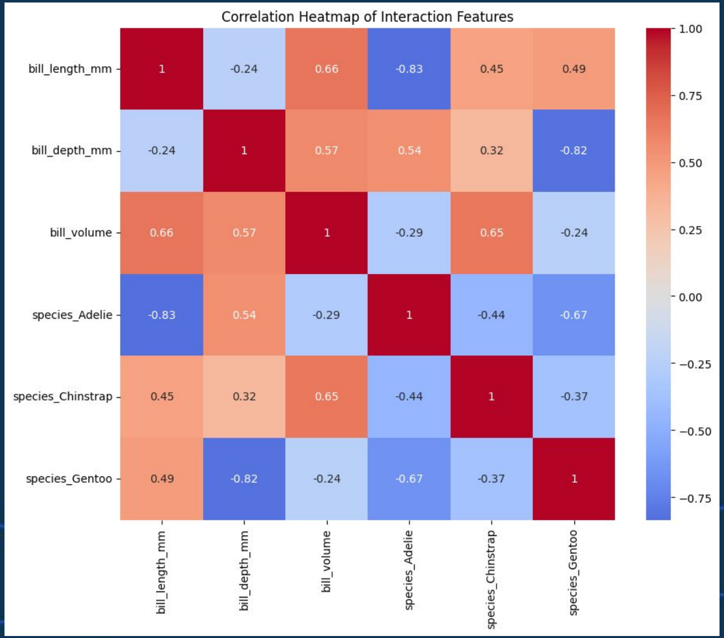
Source: [AnalyticsVidhya]

Techniques

- **Interaction features:** Combine existing features to capture interactions
- **Polynomial features:** Generate higher-order terms to capture non-linear relationships
- **Domain-specific features:** Apply domain knowledge to create meaningful features

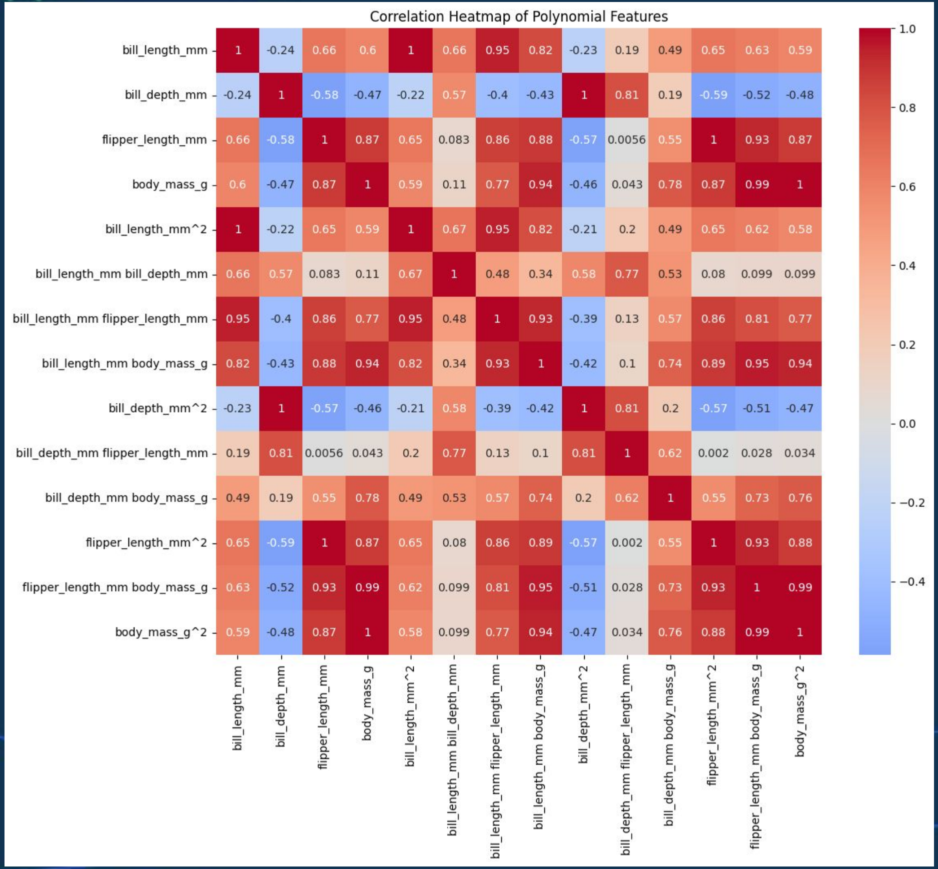
Interaction Features

```
# Interaction features
penguins['bill_volume'] = penguins['bill_length_mm'] * penguins['bill_depth_mm']
```



	bill_length_mm	bill_depth_mm	bill_volume	species_Adelie	species_Chinstrap	species_Gentoo
bill_length_mm	1	-0.24	0.66	-0.83	0.45	0.49
bill_depth_mm	-0.24	1	0.57	0.54	0.32	-0.82
bill_volume	0.66	0.57	1	-0.29	0.65	-0.24
species_Adelie	-0.83	0.54	-0.29	1	-0.44	-0.67
species_Chinstrap	0.45	0.32	0.65	-0.44	1	-0.37
species_Gentoo	0.49	-0.82	-0.24	-0.67	-0.37	1

Polynomial Features



HyperionDev

Challenge

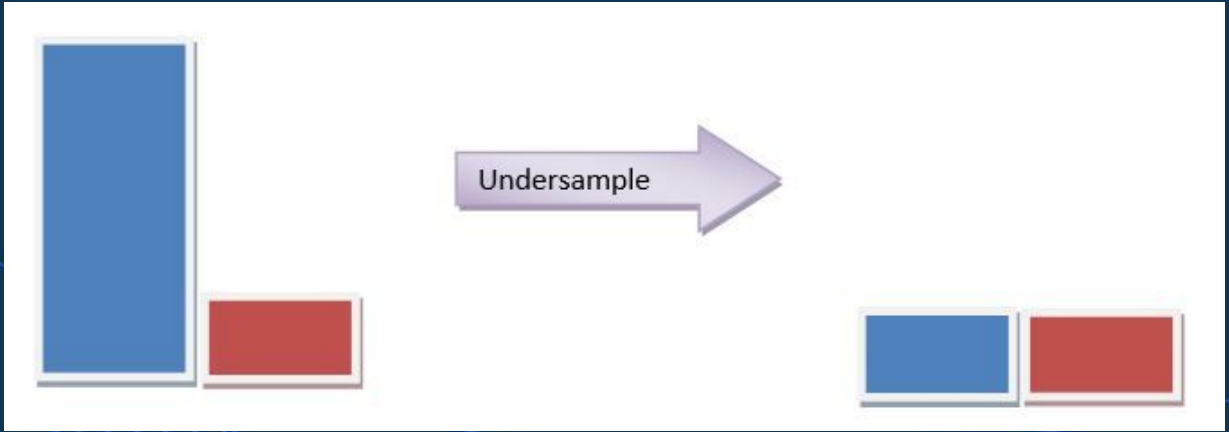
- Skewed class distribution leads to biased models and poor minority class performance

Source: [AnalyticsVidhya]

HyperionDev

Techniques

- ❖ **Undersampling:** Reduce majority class instances
  - ➢ **Random undersampling:** Remove majority instances

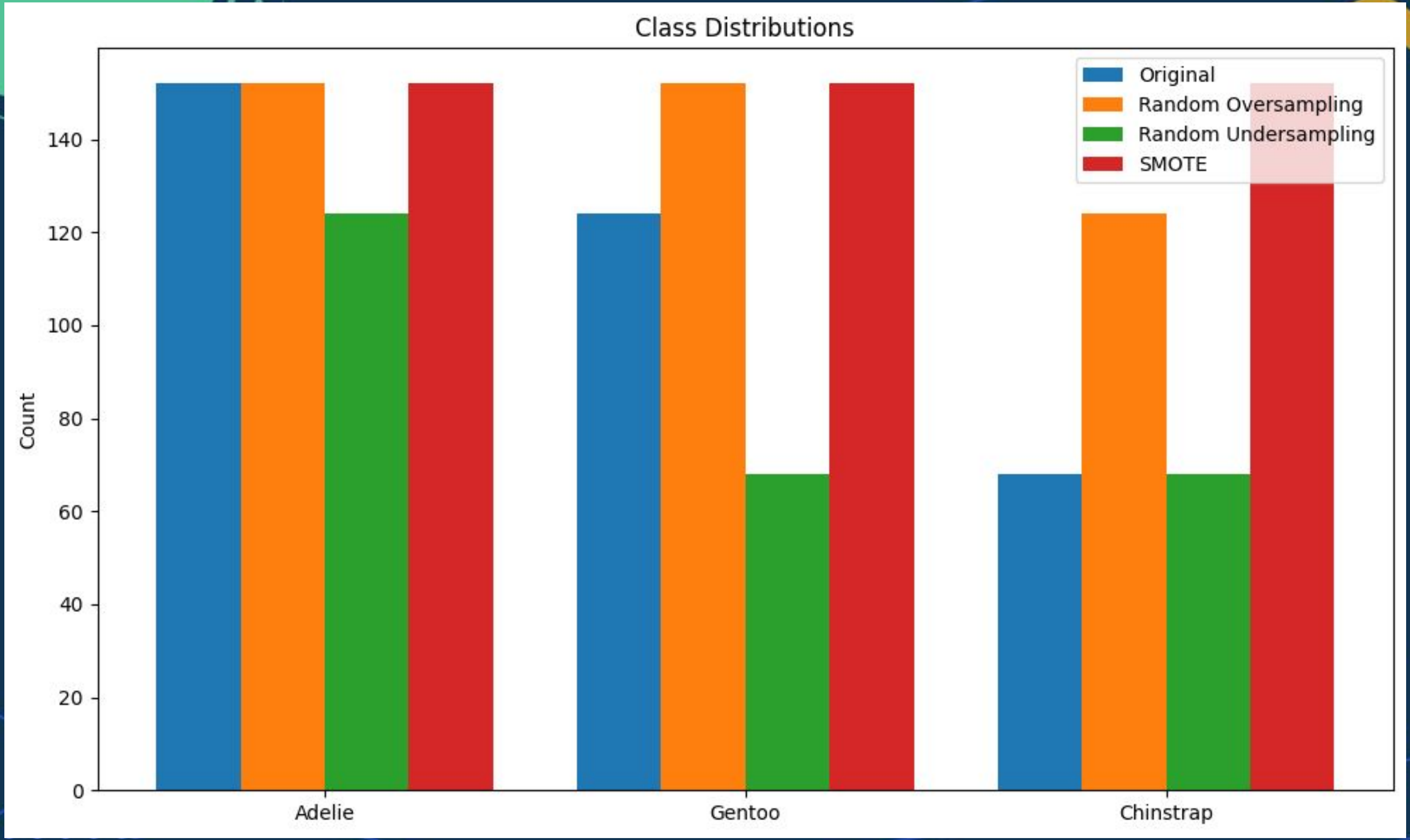


Source: [AnalyticsVidhya]

Techniques

- **Oversampling:** Increase minority class instances
  - **Random oversampling:** Duplicate minority instances
  - **SMOTE:** Generate synthetic minority instances

Source: [AnalyticsVidhya]



HyperionDev

Considerations

- Oversampling may lead to overfitting, especially with random oversampling
- Undersampling may discard potentially useful data

Introduction to EDA

- **Definition:** Exploratory Data Analysis (EDA) is the process of investigating and understanding a dataset through visual and statistical techniques.
- **Importance:** EDA is a crucial first step in any data science project as it helps uncover patterns, anomalies, and relationships in the data, guiding further analysis and decision-making.

CoGrammar

Introduction to EDA

- **Role in the data science workflow:** EDA is performed after data collection and before model building and evaluation. It helps in understanding the data, identifying data quality issues, and selecting relevant features for modeling.

CoGrammar

Simple EDA Framework

➢ **Explore Relationships:**

- Analyse relationships between features and the target variable
- Use visualisations like scatter plots, pair plots, and correlation matrices
- Identify patterns, trends, and clusters in the data

Simple EDA Framework

➢ **Assess Feature Importance:**

- Determine the significance of features using statistical tests
- Use techniques like Decision Trees or Random Forests to evaluate feature importance
- Select relevant features based on their importance and domain knowledge

Simple EDA Framework

- **Iterate and Refine:**
  - Iterate on the analysis based on the insights gained
  - Refine the data cleaning and preprocessing steps if necessary
  - Consider additional visualisations or techniques to deepen the understanding of the data

CoGrammar

Loading and Exploring the Dataset

- After loading the dataset, we'll explore its basic properties:
  - **Shape of the dataset:** number of rows and columns
  - **Features:** the independent variables in the dataset
  - **Target variable:** the dependent variable (depends on the features) we want to predict or analyse

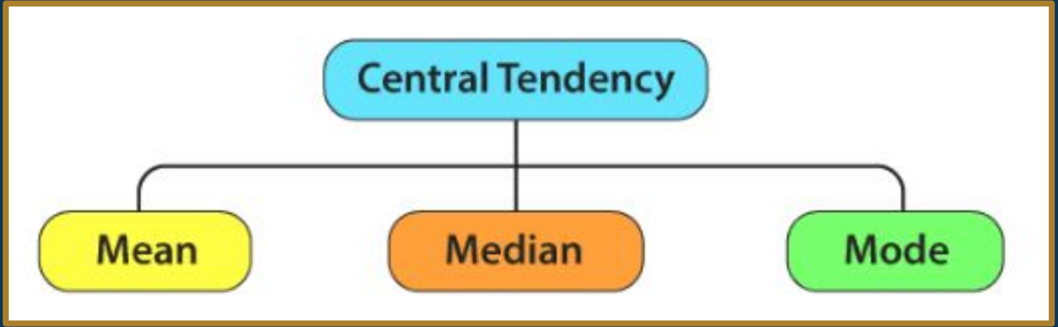
CoGrammar

Univariate Analysis

- Univariate analysis involves analysing each variable individually.
  - **Univariate:** involving one variate or variable quantity.
- We'll start by calculating descriptive statistics for the numeric variables using the describe() function.
- Descriptive statistics provide a summary of the central tendency, dispersion, and shape of the data.

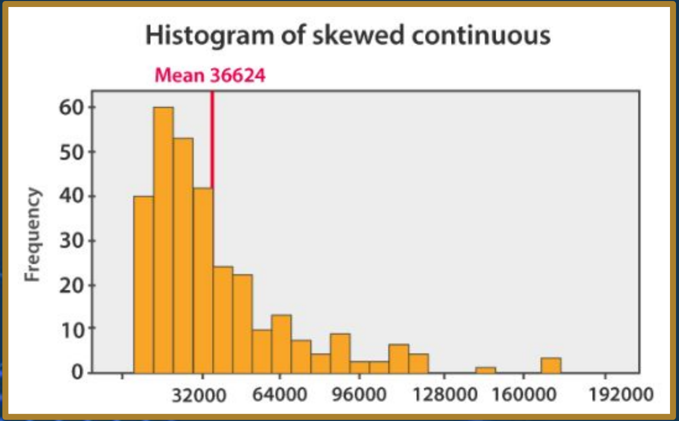
CoGrammar

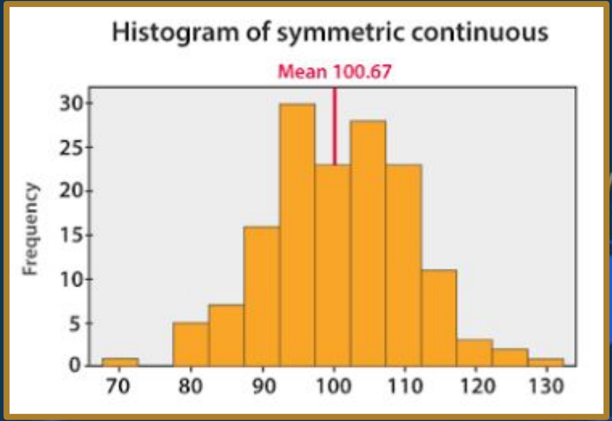
Measures of Central Tendency



Measures of Central Tendency

❖ **Mean** represents the average value of the dataset. It can be calculated as the sum of all the values in the dataset divided by the number of values.





Measures of Central Tendency

- **Median** is the middle value of the dataset in which the dataset is arranged in the ascending order or in descending order. When the dataset contains an even number of values, then the median value of the dataset can be found by taking the mean of the middle two values.

Median odd
23
21
18
16
15
13
12
10
9
7
6
5
2

Median even
40
38
35
33
32
30
29
27
26
24
23
22
19
17

28

CoGrammar

Measures of Central Tendency

- **Mode** represents the frequently occurring value in the dataset. Sometimes the dataset may contain multiple modes and in some cases, it does not contain any mode at all.

Mode
5
5
5
4
4
3
2
2
1

Univariate Analysis

- Next, we'll visualise the distribution of each feature using histograms and box plots.
- **Histograms** show the frequency distribution of a variable, helping to identify the shape, central tendency, and spread of the data.
- **Box plots** provide a summary of the distribution, highlighting the median, quartiles, and outliers.

Univariate Analysis

- We'll also check for missing values in the dataset using the isnull().sum() function.
- Missing values can impact the analysis and need to be handled appropriately.
  - Common strategies include filling missing values with the mean, median, or mode.
  - It’s usually not a good idea to just drop data, as this could skew the data and thus the results of prediction.

Bivariate Analysis

- Bivariate analysis involves examining the relationship between two variables.
    - **Bivariate:** involving or depending on two variates.
- We'll use scatter plots to visualise the relationship between features and the target variable.
- **Scatter plots** help identify patterns, correlations, and clusters in the data.

Bivariate Analysis

- To quantify the relationship between numeric features, we'll calculate the correlation matrix.
- The correlation matrix shows the pairwise correlation coefficients between variables.
- We'll visualise the correlation matrix using a heatmap.

Bivariate Analysis

❖ Interpreting the correlation matrix:

➢ Correlation coefficients range from -1 to 1.

➢ The high positive correlations between Petal Length and Petal Width (0.96) and between Sepal Length and Petal Length (0.87) suggest that these pairs of features are strongly related and may provide similar information.

➢ The low correlations between Sepal Width and the other features indicate that Sepal Width provides relatively independent information compared to the other features.

Multivariate Analysis - PCA

- Multivariate analysis involves examining relationships among multiple variables simultaneously.
- **Principal Component Analysis (PCA)** is a dimensionality reduction technique that transforms the original features into a new set of uncorrelated features called principal components.
- PCA helps identify patterns and structure in high-dimensional data by finding the directions of maximum variance.

CoGrammar

Multivariate Analysis - K-means Clustering

- K-means clustering is an unsupervised learning algorithm that partitions the data into K clusters based on similarity.
- It aims to minimize the within-cluster sum of squares (WCSS) or the Euclidean distance between data points and their cluster centroids.
- **More about this in later lectures.**

CoGrammar

Task Walkthrough

Imagine you're a data scientist at a healthcare startup. Your team is developing a predictive model to identify patients at risk of diabetes based on their medical history. Your job? Clean, explore, and analyze the data, then build a model to make predictions!

- **Linear Regression: Predict blood sugar level based on age and BMI.**
- **Logistic Regression: Classify whether a patient is at risk of diabetes.**

By the end of this task, you'll be able to transform raw data into meaningful insights and predictive models, just like a real-world data scientist! 🚀

Which of the following is a common method for handling missing values?

A. Deleting the entire dataset
B. Replacing missing values with the mean, median, or mode
C. Randomly assigning new values
D. Ignoring the missing values completely

What is one way to detect relationships between variables during Exploratory Data Analysis?

A. Using a confusion matrix
B. Creating a heatmap of correlations
C. Running a logistic regression model
D. Dropping all missing values

Summary

★ **Data Cleaning:**
Converting text to lowercase and removing whitespace for consistency.
Standardizing categorical values to ensure uniform data representation.
Handling missing values using appropriate imputation methods.
Identifying and removing duplicate records to maintain data integrity.

★ **Data Preprocessing:**
Understanding the difference between standardization and normalization:
One-hot encoding for categorical variables without inherent order.
Label encoding for ordinal categorical variables.

★ **Exploratory Data Analysis (EDA):**
Computing summary statistics using .describe().
Visualizing data distributions using histograms and boxplots to detect outliers.
Analyzing correlations with heatmaps to identify relationships.

CoGrammar

Q & A SECTION

Please use this time to ask any questions relating to the topic, should you have any.

Thank you for attending

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education