Welcome to the CoGrammar Data Preprocessing

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 Preprocessing

April 2024

Learning objectives

- Understand the importance and purpose of **data preprocessing** in data science projects
- Learn and apply **advanced data preprocessing techniques** beyond basic data cleaning
- Gain hands-on experience using **Python libraries for preprocessing real-world datasets**
- **Integrate preprocessing techniques** into machine learning workflows

CoGrammar

Introduction to Data Preprocessing

CoGrammar

Data Preprocessing

- Data preprocessing is a crucial step in any data science project, **going beyond basic data cleaning** to **ensure the quality, reliability, and suitability of data** for machine learning algorithms.

- Preprocessing techniques **address complex data issues**, such as **missing values, outliers, inconsistencies, and irrelevant features,** which can significantly impact the performance and generalisability of machine learning models.

CoGrammar

Data Preprocessing

- In this lesson, we will explore **advanced preprocessing techniques**, their importance, and how to apply them using Python libraries, equipping you with the skills to tackle real-world data challenges and improve the effectiveness of your machine learning projects.

Recap of
Data Cleaning

CoGrammar

Recap of Data Cleaning Concepts

- Data cleaning is the process of **identifying and addressing data quality issues to ensure the accuracy, completeness, and consistency** of the dataset. It is a foundational step in data preprocessing that focuses on the following key aspects:

  - **Handling missing values:** Identifying and addressing missing data points in the dataset, either by removing instances with missing values (deletion) or imputing missing values based on statistical measures or domain knowledge (imputation).

CoGrammar

Recap of Data Cleaning Concepts

- **Dealing with outliers:** Detecting and handling outliers, which are data points that significantly deviate from the majority of the data. Outliers can be identified using statistical methods, such as Z-score or Interquartile Range (IQR), and addressed by removal, transformation, or using robust models.

Recap of Data Cleaning Concepts

- **Resolving inconsistencies:** Identifying and resolving inconsistencies in data format, units, or conventions across different features or data sources. This ensures that the data is **standardised and comparable** for analysis and modeling.

Recap of Data Cleaning Concepts

- **Removing duplicates:** Identifying and removing duplicate records from the dataset to **avoid redundancy and potential biases in the analysis.** Duplicates can be exact matches or near-duplicates based on specific criteria.

CoGrammar

Recap of Data Cleaning Concepts

❖ **Transforming data types:** Converting data types to ensure compatibility with the requirements of machine learning algorithms and to optimise computational efficiency. This includes **converting categorical variables to numerical representations and handling data type mismatches.**

CoGrammar

Importance of Data Preprocessing

CoGrammar

Importance of Data Preprocessing

- **Improved data quality:** Advanced preprocessing techniques help address data issues that **may not be captured by basic cleaning, such as handling missing data patterns, feature scaling, and encoding categorical variables.** By enhancing the quality and consistency of the data, preprocessing **lays the foundation for accurate and reliable machine learning models.**

CoGrammar

Importance of Data Preprocessing

- **Enhanced model performance:** Preprocessed data is optimised for machine learning algorithms, **enabling them to extract meaningful patterns and relationships more effectively.** Techniques like **feature scaling, encoding, and handling imbalanced data** can significantly improve model **accuracy, generalisation, and robustness**.

CoGrammar

Importance of Data Preprocessing

❖ **Reduced computational complexity:** Preprocessing techniques can help **reduce the dimensionality of the data by selecting relevant features, transforming variables, and creating more efficient representations.** This leads to faster model training and inference times, making the machine learning process more computationally efficient.

CoGrammar

Overview

CoGrammar

Overview

- **Handling missing data:** Identifying missing data patterns and mechanisms, such as **Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR).** Applying advanced imputation techniques, such as K-Nearest Neighbors (KNN) imputation, Multiple Imputation by Chained Equations (MICE), and matrix factorisation, to estimate missing values based on patterns in the available data.

- This process is done in either/both data cleaning and preprocessing.

Overview

- **Feature scaling:** Transforming features to **a common scale to ensure fair comparison and contribution in the machine learning process.** Techniques include **standardisation (Z-score normalisation), min-max scaling, and robust scaling**, which handle different data distributions and outliers.

CoGrammar

Overview

- **Encoding categorical variables:** Converting **categorical variables into numerical representations** suitable for machine learning algorithms. Techniques include **one-hot encoding, label encoding, ordinal encoding, and binary encoding,** each addressing different types of categorical variables (nominal or ordinal) and handling high-cardinality scenarios.

CoGrammar

Overview

- **Feature engineering:** Creating new features from existing data to capture domain knowledge, underlying patterns, and relationships. Techniques include **polynomial features, interaction features, and domain-specific transformations**.

Overview

- **Handling imbalanced data:** Addressing the challenge of imbalanced class distributions in classification tasks, **where one class (minority class) is significantly underrepresented compared to others.** Techniques include **oversampling (random oversampling, SMOTE), undersampling (random undersampling, Tomek links), and class weight adjustment, which help balance the class distribution and improve model performance on the minority class.**

CoGrammar

Feature Scaling

CoGrammar

Feature Scaling

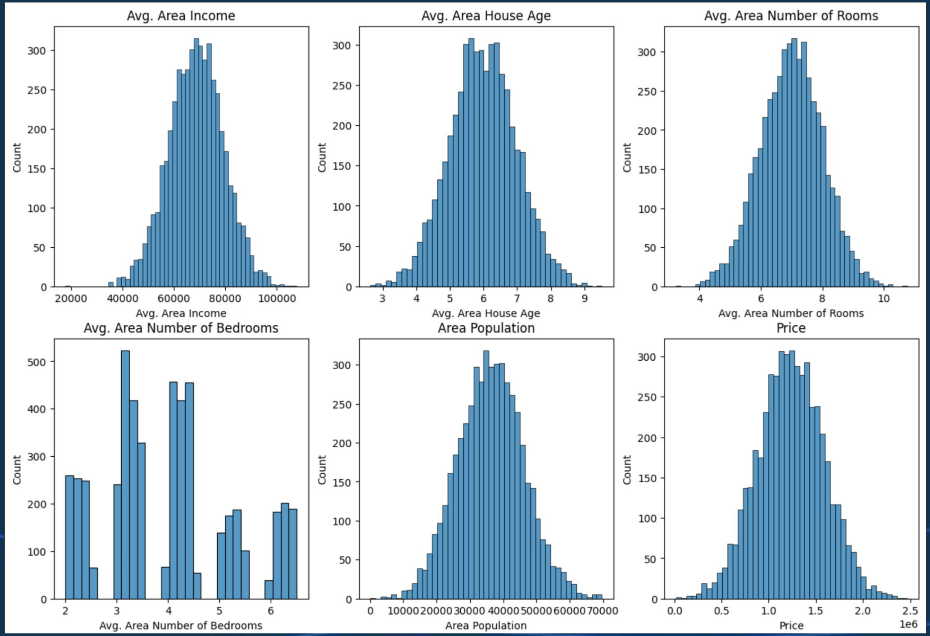
- Many machine learning algorithms are **sensitive to the scale and distribution of input features.** Features with l**arger values or wider ranges can dominate the learning process, leading to biased or suboptimal models**. Feature scaling ensures that all features contribute equally to the model, improving convergence, stability, and interpretability.

Feature Scaling

- For example, in a customer churn prediction model, features like "total_spent" (ranging from **100 to 10000**) and "age" (ranging from **18 to 80**) have different scales. Scaling these features to a common range (e.g., **0 to 1**) ensures that **the model gives equal importance to both features during training.**

CoGrammar

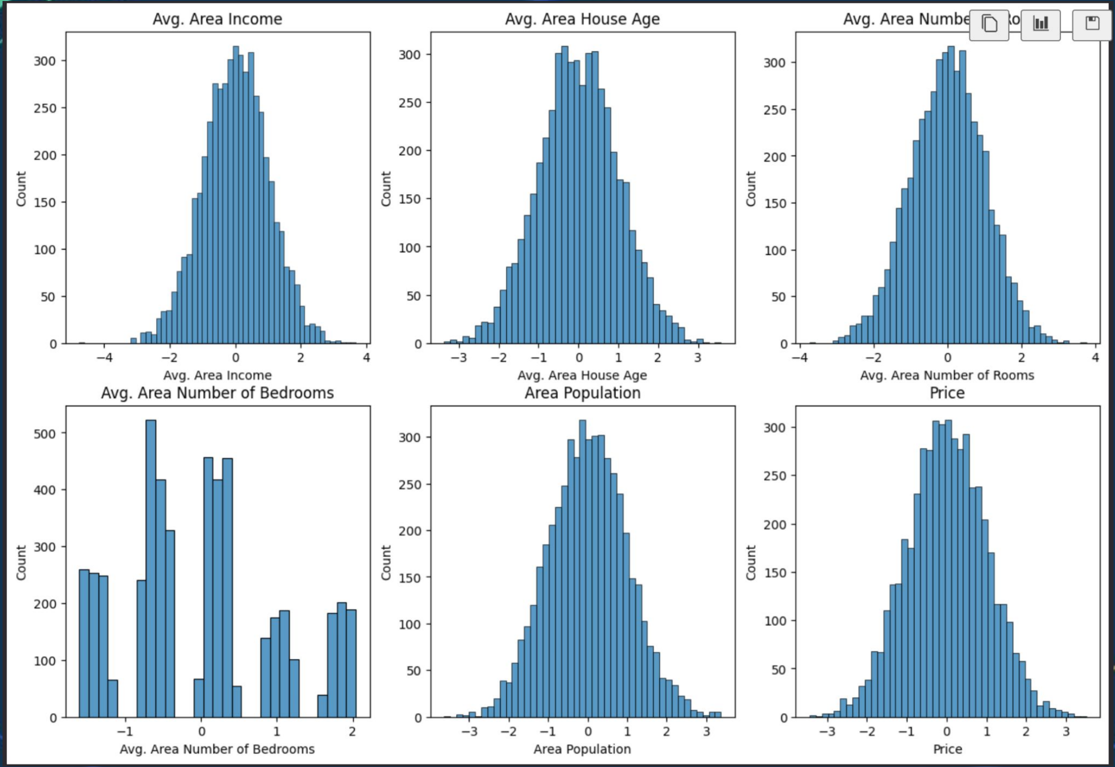
❖ Let’s take these features as examples to explain the upcoming techniques:



CoGrammar

Techniques for feature scaling

- **Standardisation (Z-score normalisation):** This technique transforms features to have **zero mean and unit variance.** It is useful when the data follows a Gaussian distribution and is sensitive to outliers. The scaled values represent the number of standard deviations away from the mean.
  - **Formula:** (x - mean) / standard deviation

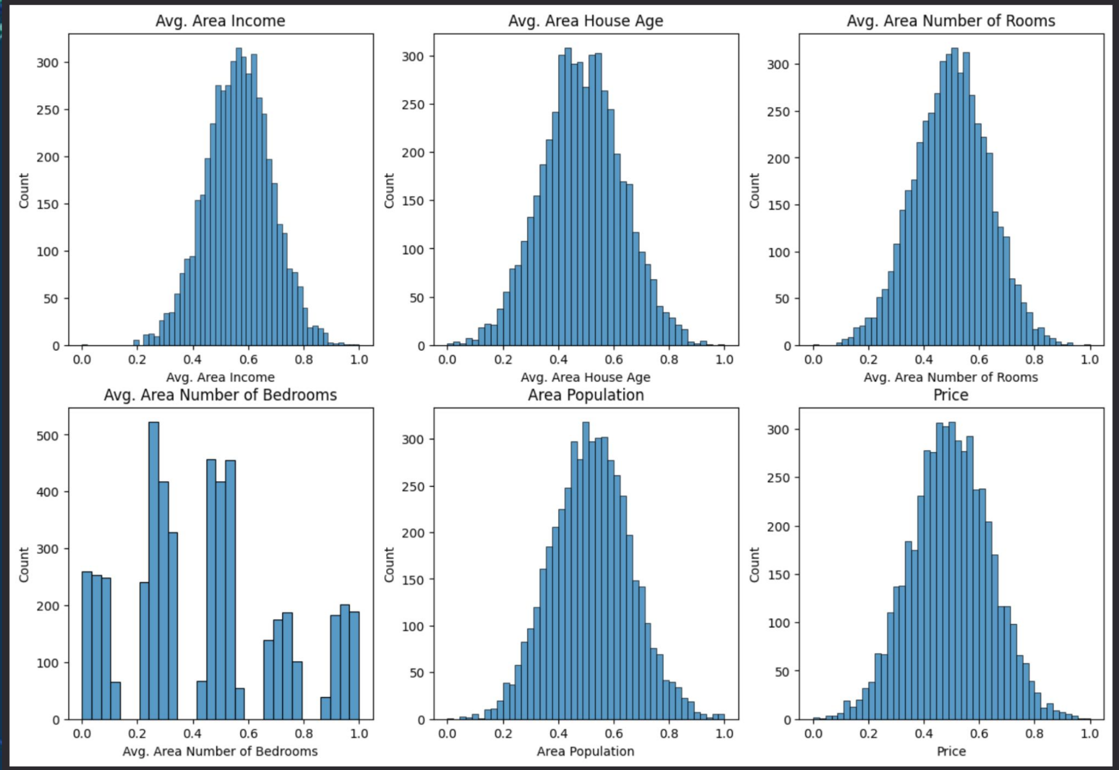

Avg. Area Income
Count
Avg. Area Income
Avg. Area House Age
Count
Avg. Area House Age
Count
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Count
Avg. Area Number of Bedrooms
Area Population
Count
Area Population
Price
Count
Price

CoGrammar

Techniques for feature scaling

- **Min-Max scaling:** This technique **scales features to a specific range, typically 0 to 1.** It is useful when the data does not follow a Gaussian distribution or when bounded features are required. The scaled values represent the relative position within the original range.
  - **Formula:** (x - min) / (max - min)

CoGrammar

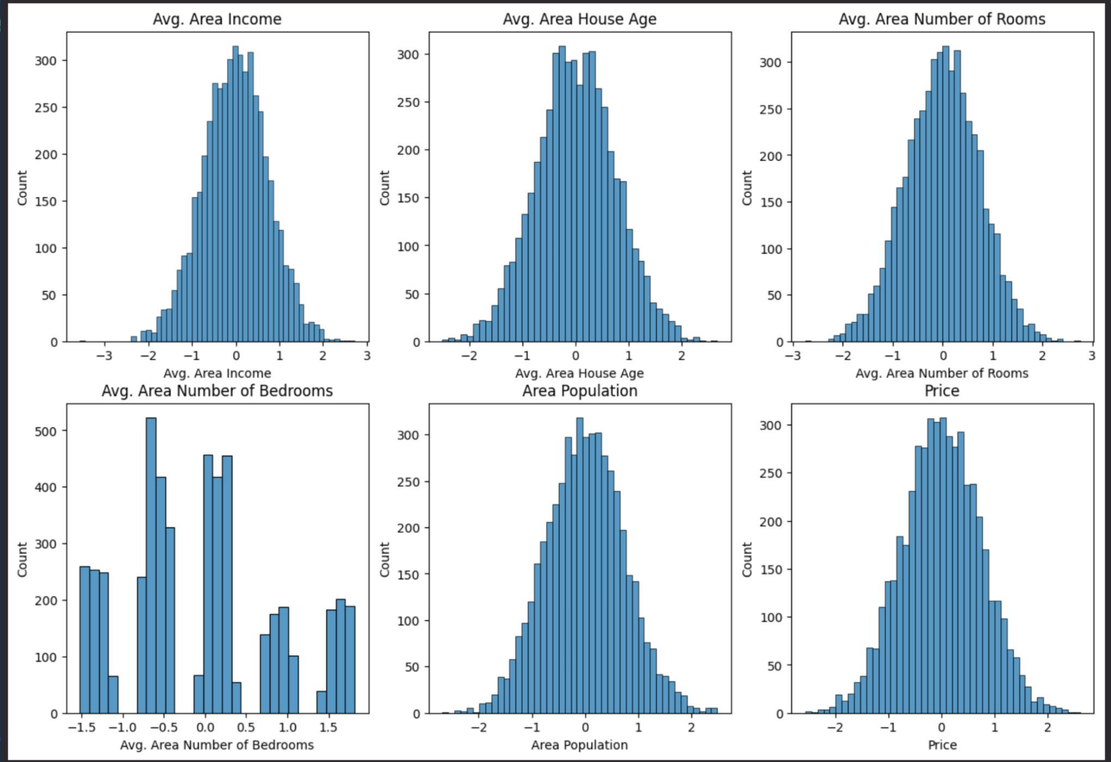

Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
Price
Count
0.0 0.2 0.4 0.6 0.8 1.0
Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
Price

CoGrammar

Techniques for feature scaling

- **Robust scaling:** This technique uses robust statistics (median and interquartile range) to scale features, making it less sensitive to outliers. It is useful **when the data contains outliers that should not significantly impact the scaling process**.
  - Formula: (x - median) / IQR

CoGrammar



Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
Price
Count
Avg. Area Income
Avg. Area House Age
Avg. Area Number of Rooms
Avg. Area Number of Bedrooms
Area Population
Price

CoGrammar

Choosing the appropriate scaling technique

- Consider the characteristics of your data, such as distribution, outliers, and domain knowledge. **Standardisation is often a good default choice, while Min-Max scaling is suitable for bounded features or non-Gaussian data.** R**obust scaling is recommended when outliers are present but should not have a large impact on the scaled values.**

CoGrammar

Choosing the appropriate scaling technique

❖ Experiment with different scaling techniques and **evaluate their impact on model performance**. The choice of scaling technique can be included in a hyperparameter tuning process to find the optimal preprocessing strategy for your specific problem.

CoGrammar

What is the purpose of feature scaling?

A. To convert categorical variables into numerical representations

B. To create new features from existing data

C. To ensure fair comparison and contribution of features in machine learning

D. To handle imbalanced class distributions

CoGrammar

What is the purpose of feature scaling?

A. To convert categorical variables into numerical representations

B. To create new features from existing data

C. To ensure fair comparison and contribution of features in machine learning

D. To handle imbalanced class distributions

CoGrammar

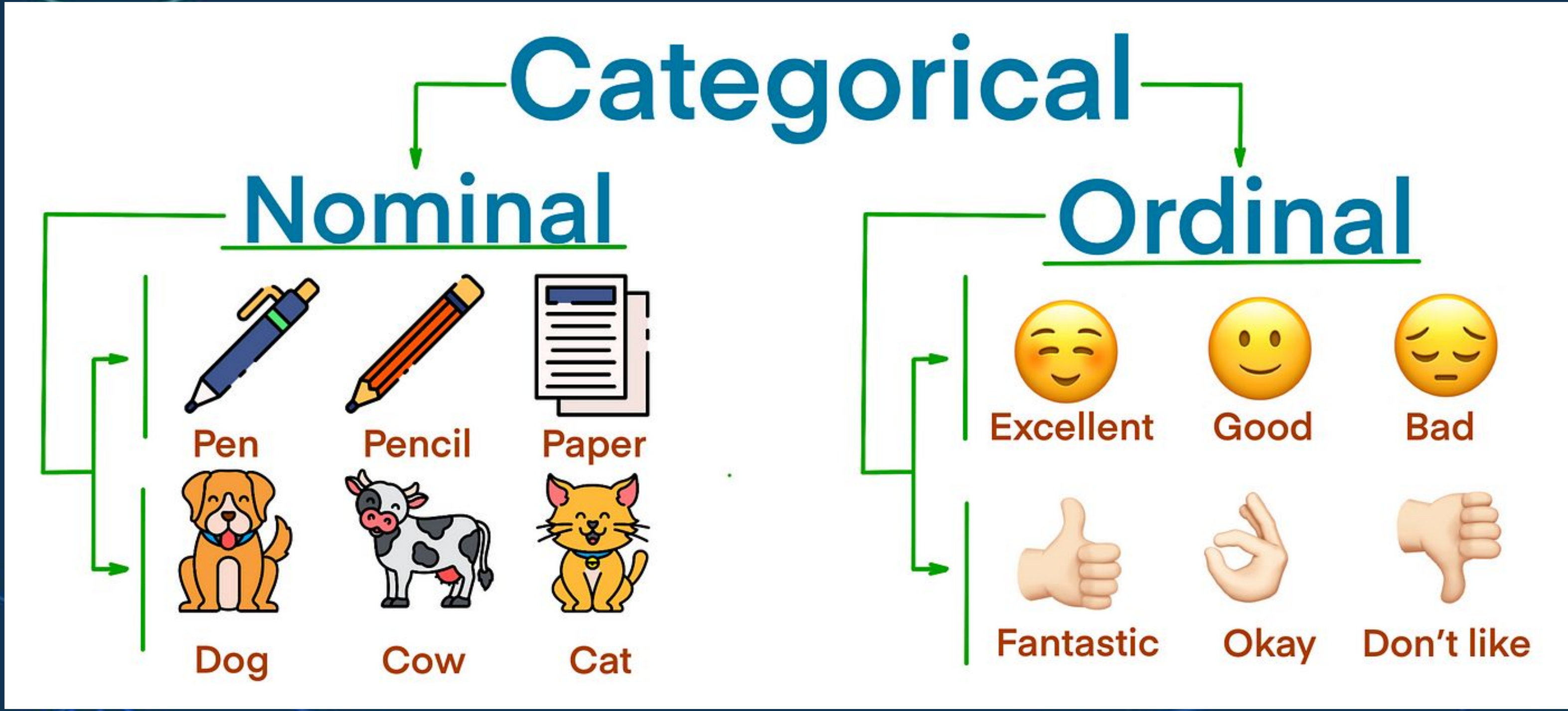
Encoding Categorical Variables

CoGrammar

Nominal vs. Ordinal

- Nominal variables have categories **without an inherent order or ranking.** Examples include "color" (red, blue, green) or "gender" (male, female). The numerical encoding should not imply any ordinal relationship between the categories.

- Ordinal variables have categories with a **meaningful order or ranking.** Examples include "size" (small, medium, large) or "education level" (high school, bachelor's, master's, PhD). The numerical encoding should preserve the ordinal information.


Categorical
Nominal
Pen
Pencil
Paper
Dog
Cow
Cat
Ordinal
Excellent
Good
Bad
Fantastic
Okay
Don't like

CoGrammar

Encoding nominal variables

- **Binary Encoding:** This technique **encodes categories as binary integers,** reducing the dimensionality compared to one-hot encoding. **It assigns a unique binary code to each category. Binary encoding is useful when the number of categories is large, and the increased dimensionality from one-hot encoding is a concern.**

	Sex_female	Sex_male
0	0.0	1.0
1	1.0	0.0
2	1.0	0.0
3	1.0	0.0
4	0.0	1.0

Encoding ordinal variables

- **Label Encoding:** This technique assigns numerical labels to categories **based on their order**. It is suitable for ordinal variables with a meaningful order. However, l**abel encoding implies a linear relationship between the encoded values, which may not always be appropriate.**

Height
Tall
Medium
Short

Height
0
1
2

High-cardinality variables

- When dealing with categorical variables with a **large number of unique categories (high cardinality),** traditional encoding techniques like one-hot encoding can lead to a significant increase in dimensionality. In such cases, alternative techniques can be employed:

  - **Frequency-based encoding:** Replacing categories with their frequency or occurrence count in the dataset. This technique captures the relative importance of each category.

High-cardinality variables

- When dealing with categorical variables with a **large number of unique categories (high cardinality),** traditional encoding techniques like one-hot encoding can lead to a significant increase in dimensionality. In such cases, alternative techniques can be employed:

  - **Target encoding:** Replacing categories with the mean or median of the target variable for each category. This technique incorporates the relationship between the categorical variable and the target variable.

CoGrammar

High-cardinality variables

- When dealing with categorical variables with a **large number of unique categories (high cardinality),** traditional encoding techniques like one-hot encoding can lead to a significant increase in dimensionality. In such cases, alternative techniques can be employed:

    - **Hashing:** Applying a hash function to the categories and using the resulting hash values as numerical representations. This technique reduces dimensionality while preserving **some** information about the categories.

CoGrammar

What is the main difference between nominal and ordinal variables?

A. Nominal variables have categories with an inherent order, while ordinal variables do not

B. Ordinal variables have categories with a meaningful order, while nominal variables do not

C. Nominal and ordinal variables are the same

D. Nominal variables are always encoded using one-hot encoding, while ordinal variables use label encoding

What is the main difference between nominal and ordinal variables?

A. Nominal variables have categories with an inherent order, while ordinal variables do not

B. Ordinal variables have categories with a meaningful order, while nominal variables do not

C. Nominal and ordinal variables are the same

D. Nominal variables are always encoded using one-hot encoding, while ordinal variables use label encoding

CoGrammar

Feature Engineering

CoGrammar

Feature Engineering

- Feature engineering is the **process of creating new features from existing data to capture domain knowledge, underlying patterns, and relationships that can improve model performance and interpretability.** It involves transforming raw data into informative and relevant features that enhance the predictive power of machine learning models.

Importance of Feature Engineering

- **Incorporating domain expertise:** Feature engineering allows data scientists to **leverage their domain knowledge and understanding of the problem** to create meaningful features that capture important aspects of the data. By incorporating domain-specific insights, models can better capture the underlying patterns and relationships.

CoGrammar

Importance of Feature Engineering

- **Improving model performance:** Well-engineered features can significantly improve the performance of machine learning models by **providing more relevant and informative inputs.** By capturing key aspects of the data and reducing noise, feature engineering can lead to better model accuracy, generalisation, and robustness.

Importance of Feature Engineering

- **Enhancing interpretability:** Feature engineering can create features that are **more interpretable and understandable to stakeholders.** By transforming raw data into meaningful features, models can provide insights and explanations that are easier to communicate and act upon.

Feature Engineering Techniques

- **Interaction features:** This technique creates new features by **combining existing ones to capture interactions and dependencies between features.** Interaction features can reveal how the combined effect of multiple features influences the target variable.

```
housing_interact[
    'Avg. Area Bedrooms per Room'
    ] = housing_interact[
        'Avg. Area Number of Bedrooms'
        ] / housing_interact[
            'Avg. Area Number of Rooms'
            ]
```

CoGrammar

Feature Engineering Techniques

- **Domain-specific features:** This technique involves applying domain expertise to **create features that are specific to the problem at hand.** Domain knowledge can help identify relevant factors, derived variables, and aggregations that are meaningful for the given context.

  - Example: In a retail sales prediction problem, creating features like "days_since_last_purchase", "average_basket_value", or "product_category_preferences" based on domain understanding

CoGrammar

Practical considerations

- It is important to **validate** the effectiveness of engineered features through **model evaluation and feature importance analysis.** By assessing the impact of each feature on model performance, data scientists can refine and iterate on their feature engineering approach to create the most informative and relevant features for the problem at hand.

- We'll look at this when we get to model evaluation.

Imbalance Data

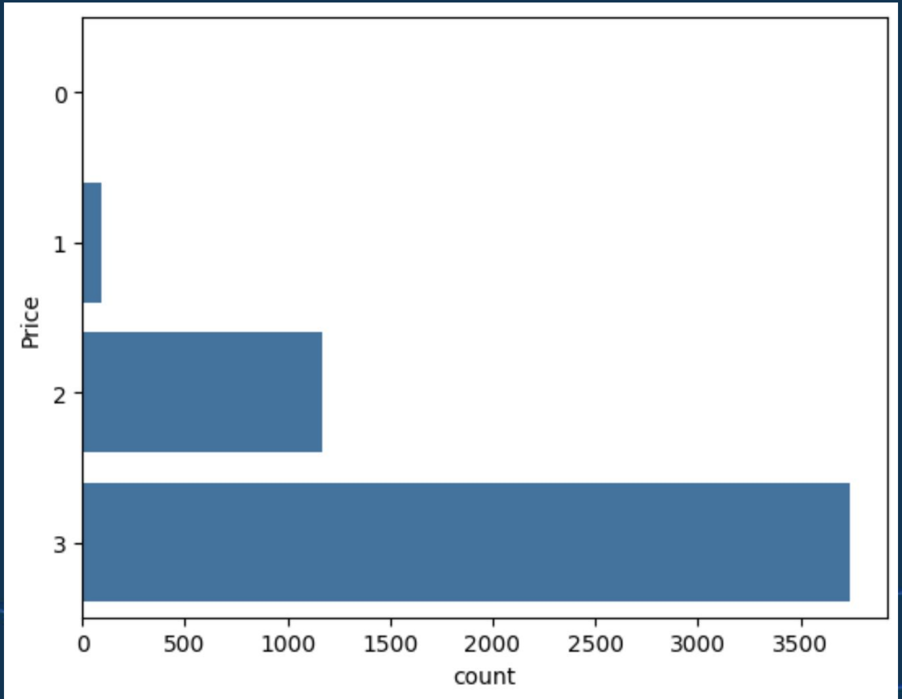
CoGrammar

Imbalanced Data

- Imbalanced datasets occur when the **distribution of classes in the target variable is highly skewed, with one or more classes being significantly underrepresented compared to others.** Imbalanced data poses challenges for machine learning algorithms, as they tend to be biased towards the majority class, leading to poor performance on the minority class.

CoGrammar

Imbalanced Data



Handling Imbalanced Data

- **Avoiding biased models:** When trained on imbalanced data, machine learning models can become biased towards the majority class, as they optimise for overall accuracy. This bias leads to **high accuracy on the majority class but poor performance on the minority class,** which is often the class of interest.

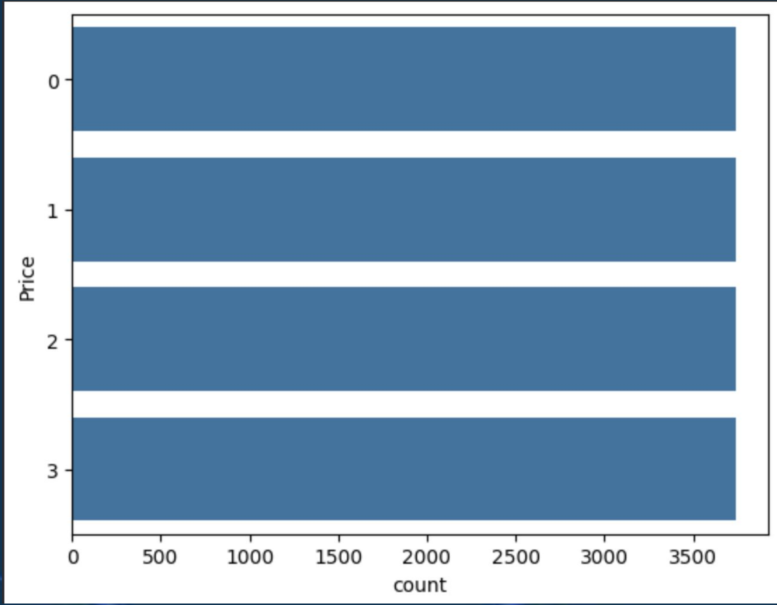
Techniques to Handle Imbalanced Data

- **Oversampling:** This technique involves increasing the number of instances in the minority class to balance the class distribution. Oversampling can be performed by duplicating existing minority instances or generating synthetic instances.

  - **Random oversampling:** Randomly duplicating minority class instances until the desired class balance is achieved.

CoGrammar

Techniques to Handle Imbalanced Data



CoGrammar

Techniques to Handle Imbalanced Data

- **Synthetic Minority Over-sampling Technique (SMOTE):** Generating synthetic minority instances by interpolating between existing instances in the feature space.

CoGrammar

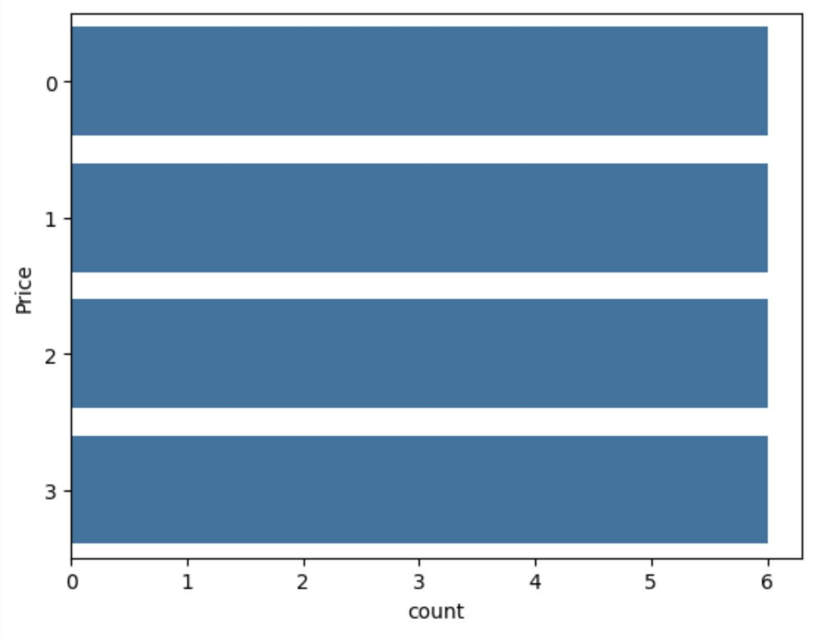
Techniques to Handle Imbalanced Data

- **Undersampling:** This technique involves **reducing the number of instances in the majority class to balance the class distribution**. Undersampling can be performed by randomly removing majority instances or using targeted approaches.

    - **Random undersampling:** Randomly removing majority class instances until the desired class balance is achieved.

CoGrammar

Techniques to Handle Imbalanced Data



CoGrammar

What is the purpose of oversampling in handling imbalanced data?

A. To reduce the number of instances in the majority class

B. To remove majority instances that are close to minority instances

C. To increase the number of instances in the minority class

D. To generate synthetic minority instances by interpolating between existing instances

What is the purpose of oversampling in handling imbalanced data?

A. To reduce the number of instances in the majority class

B. To remove majority instances that are close to minority instances

C. To increase the number of instances in the minority class

D. To generate synthetic minority instances by interpolating between existing instances

Conclusions

CoGrammar

Conclusions

- Recap of key points:
  - Data preprocessing is an essential step in the data science pipeline.
  - Various techniques and tools are available for effective data preprocessing.
  - We saw that involves **scaling data, encoding variables, engineering features, and handling imbalanced data.**

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar