MPSSDEGB
कौशल से रोजगार, समृद्धि अपार

Foundation For Innovation And Technology Transfer

SANKALP
Ministry of Skill Development & Entrepreneurship

DAY 8 - Data Preprocessing and Feature Engineering

.

Jan, 2024

Contents

Data Pre processing and Feature Engineering

- Data preprocessing and Feature engineering is the process of selecting, manipulating, and transforming raw data into features that can be used in supervised learning.

- Data preprocessing is a technique that is used to convert the raw data into a clean data set.

- Feature Engineering is the preprocessing step of machine learning,which extract features from raw data.

- The process helps to represent an underlying problem to predictive models in a better way, which as a result, improve the accuracy of the modelfor unseen data.

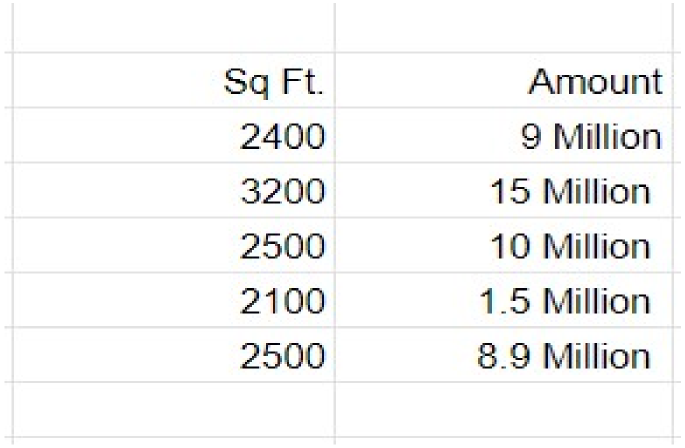
Data Pre processing and Feature Engineering

- The Predictive model contains predictor variables and outcome variable and while data preprocessing and feature engineering process selects the most useful predictor variables for the model.

- The process can produce new features for both supervised and unsupervised learning, with the goal of simplifying and speeding up data transformations while also enhancing model accuracy.

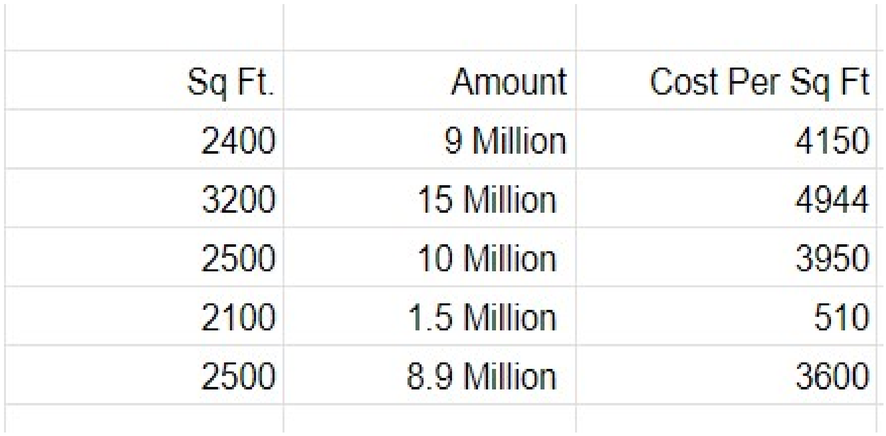
Data Pre processing and Feature Engineering

Let’s take a **simple example**. Below are the prices of properties in x city. It shows the area of the house and total price.



Sq Ft.	Amount
2400	9 Million
3200	15 Million
2500	10 Million
2100	1.5 Million
2500	8.9 Million

Fig.1



Sq Ft.	Amount	Cost Per Sq Ft
2400	9 Million	4150
3200	15 Million	4944
2500	10 Million	3950
2100	1.5 Million	510
2500	8.9 Million	3600

Fig.2

Now this data might have some errors or might be incorrect. To begin, we’ll add a new column to display the cost per square foot.

This new feature will help us understand a lot about our data. So, we have a new column which shows cost per square ft.

Data Pre processing and Feature Engineering

Data Preprocessing and Feature engineering consists of various process –

1. **Feature Creation**: Creating features involves creating new variables which will be most helpful for our model. This can be adding or removing some features. As we saw above, the cost per sq. ft column was a feature creation.

2. **Transformations**: Feature transformation is simply a function that transforms features from one representation to another. The goal here is to plot and visualise data, if something is not adding up with the new features we can reduce the number of features used, speed up training, or increase the accuracy of a certain model.

3. **Feature Extraction**: Feature extraction is the process of extracting features from a data set to identify useful information. Without distorting the original relationships or significant information, this compresses the amount of data into manageable quantities for algorithms to process.

Data Pre processing and Feature Engineering

4. Exploratory Data Analysis : Exploratory data analysis (EDA) is a powerful and simple tool that can be used to improve your understanding of your data, by exploring its properties. The technique is often applied when the goal is to create new hypotheses or find patterns in the data. It’s often used on large amounts of qualitative or quantitative data that haven’t been analyzed before.

5. Benchmark : A Benchmark Model is the most user-friendly, dependable, transparent, and interpretable model against which you can measure your own. It’s a good idea to run test datasets to see if your new machine learning model outperforms a recognised benchmark.

Data Pre processing and Feature Engineering

Importance Of Data Preprocessing:

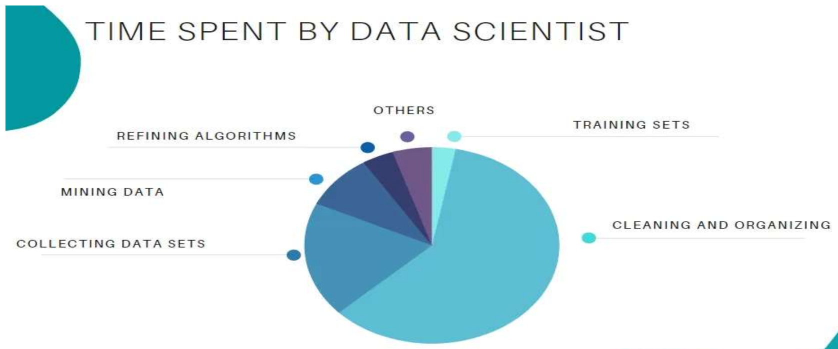
Preprocessing data is an important step for data analysis. The following are some benefits of preprocessing data:

- **Improves accuracy and reliability.** Preprocessing data removes missing or inconsistent data values resulting from human or computer error, which can improve the accuracy and quality of a dataset, making it more reliable.
- **Makes data consistent.** When collecting data, it's possible to have data duplicates, and discarding them during preprocessing can ensure the data values for analysis are consistent, which helps produce accurate results.
- **Increases the data's algorithm readability.** Preprocessing enhances the data's quality and makes it easier for machine learning algorithms to read, use, and interpret it.

Data Pre processing and Feature Engineering

Importance Of Feature Engineering:

- Feature engineeringand refers to the process of designing artificial features into an algorithm. These artificial features are then used by that algorithm in order to improve its performance, or in other words reap better results. Data scientists spend most of their time with data, and it becomes important to make models accurate.
- When feature engineering activities are done correctly, the resulting dataset is optimal and contains all of the important factors that affect the business problem. As a result of these datasets, the most accurate predictive models and the most useful insights are produced.



Data Pre processing and Feature Engineering

Data Preprocessing and Feature Engineering Techniques:

1.Data Cleaning
2.DataTransform
3. Data Reduction
4. Categorical Encoding
5. Scaling

Data Pre processing and Feature Engineering

1.Data Cleaning

Data cleaning is not just about erasing data or filling in missing values. It's a comprehensive process involving various techniques to transform raw data into a format suitable for analysis.

a) **Handling Missing Values**: Missing data can occur for various reasons, such as errors in data collection or transfer. There are several ways to handle missing data, depending on the nature and extent of the missing values

b) **Removing Duplicates**: Duplicate entries can occur for various reasons, such as data entry errors or data merging. These duplicates can skew the data and lead to biased results. Techniques for removing duplicates involve identifying these redundant entries based on key attributes and eliminating them from the dataset.

Data Pre processing and Feature Engineering

c) **Outlier Detection**: Outliers are data points that significantly deviate from other observations. They can be caused by variability in the data or errors. Outlier detection techniques are used to identify these anomalies.

Data Pre processing and Feature Engineering

a) Handling Missing Values

1.Imputation

When it comes to preparing your data for machine learning, missing values are one of the most typical issues. Human errors, data flow interruptions, privacy concerns, and other factors could all contribute to missing values. Missing values have an impact on the performance of machine learning models for whatever cause. The main goal of imputation is to handle these missing values. There are two types of imputation :

Numerical Imputation: Numerical data imputation algorithms replace missing values by estimates to leverage in- complete data sets. such as age, price, salary, and so on.

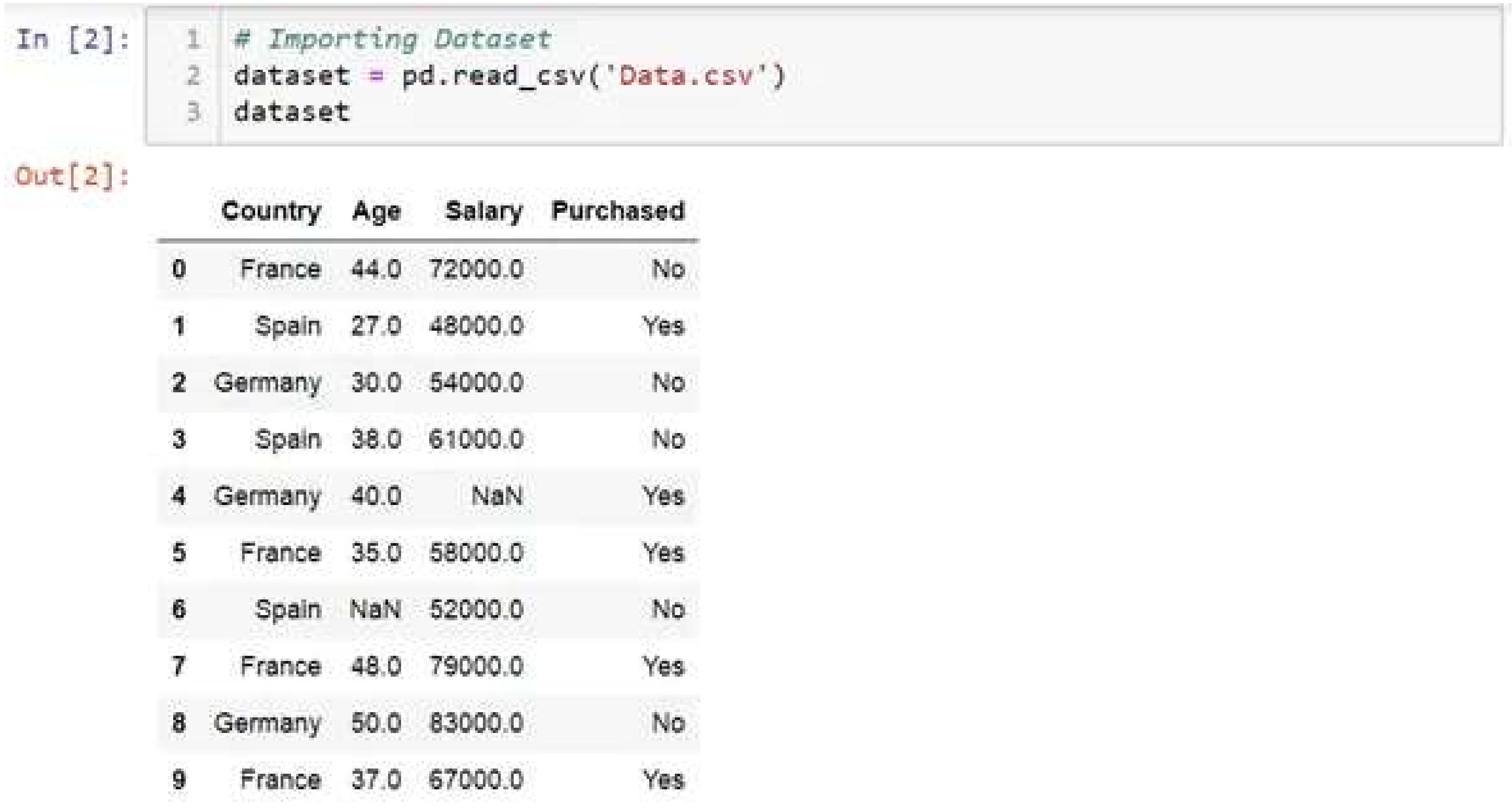
Eg.

```
#Filling all missing values with 0
data = data.fillna(0)
```

Data Pre processing and Feature Engineering

Importing the dataset

Loading the data using Pandas library using *read_csv()* method.



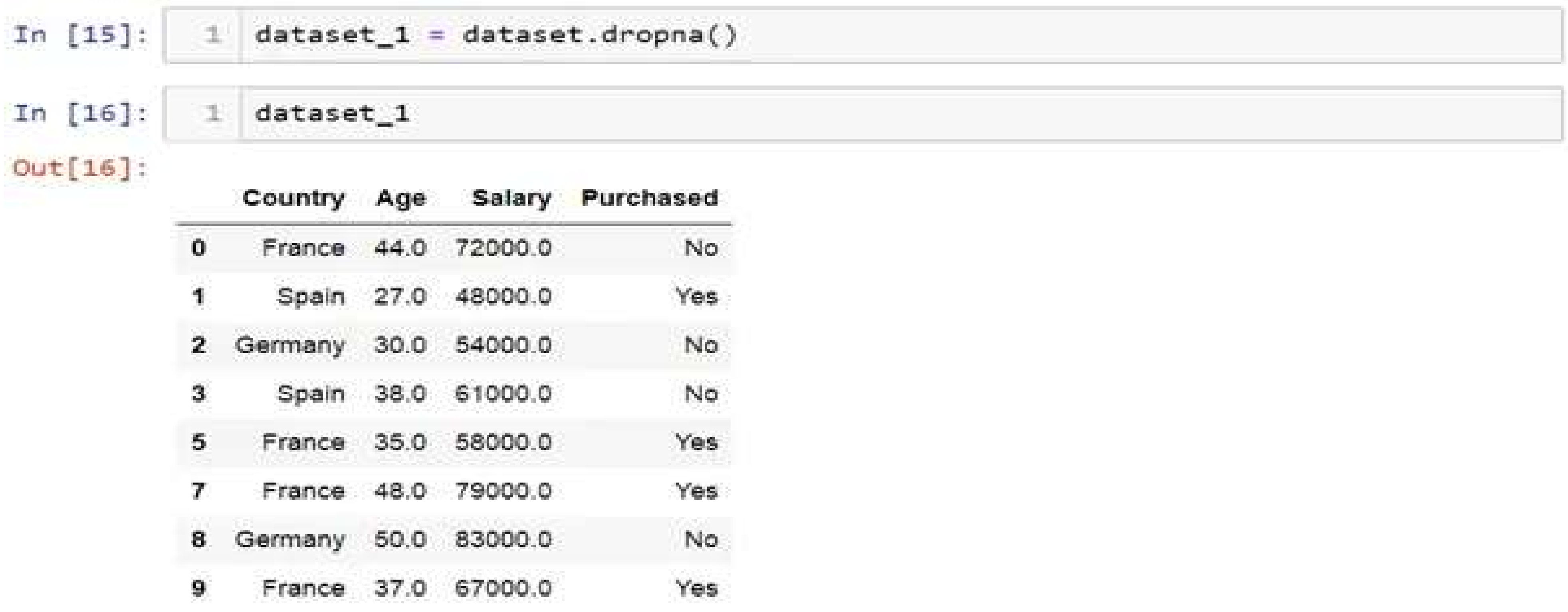
```
In [2]:  # Importing Dataset
         dataset = pd.read_csv('Data.csv')
         dataset
```

```
Out[2]:
   Country   Age   Salary  Purchased
0   France  44.0  72000.0         No
1    Spain  27.0  48000.0        Yes
2  Germany  30.0  54000.0         No
3    Spain  38.0  61000.0         No
4  Germany  40.0      NaN        Yes
5   France  35.0  58000.0        Yes
6    Spain   NaN  52000.0         No
7   France  48.0  79000.0        Yes
8  Germany  50.0  83000.0         No
9   France  37.0  67000.0        Yes
```

The 4th and 6th index have null values.

Data Pre processing and Feature Engineering

dropna() function that can be used to drop either row or columns with missing data. We can use *dropna()* to remove all the rows with missing data.



```
In [15]:  dataset_1 = dataset.dropna()
```

```
In [16]:  dataset_1
```

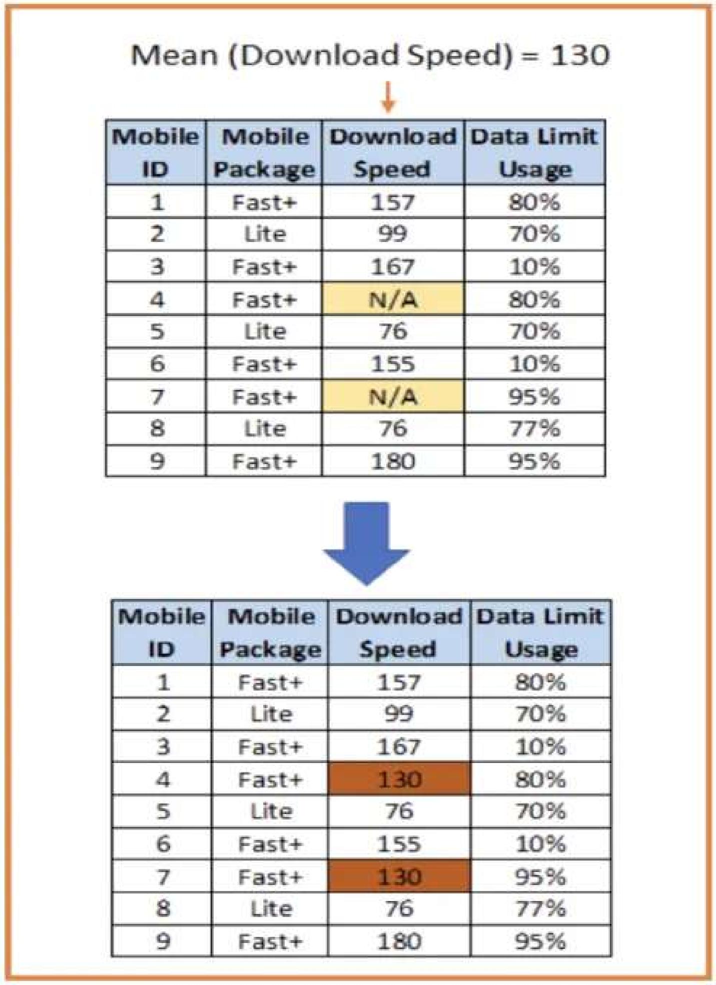
```
Out[16]:
```

	Country	Age	Salary	Purchased
0	France	44.0	72000.0	No
1	Spain	27.0	48000.0	Yes
2	Germany	30.0	54000.0	No
3	Spain	38.0	61000.0	No
5	France	35.0	58000.0	Yes
7	France	48.0	79000.0	Yes
8	Germany	50.0	83000.0	No
9	France	37.0	67000.0	Yes

We can see before there is 4th and 6th index have null values. Now as above we can see both rows with missing data has been removed.

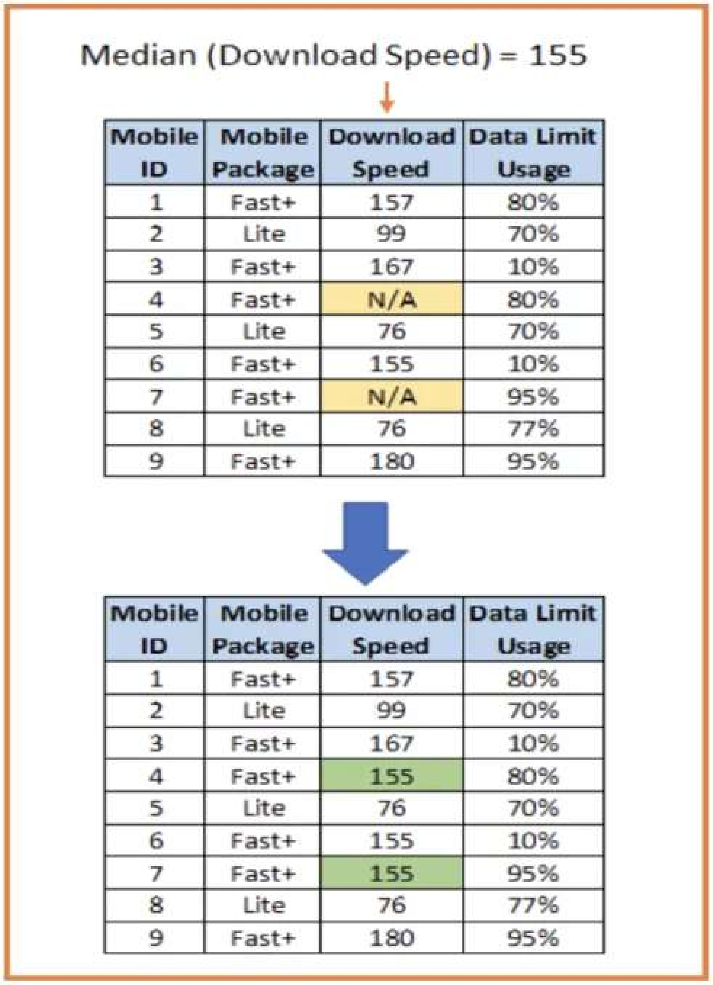
Data Pre processing and Feature Engineering

#Replacing Null values with Strategy:
For replacing null values we use the strategy that can be applied on a feature which has numeric data. We can calculate the *Mean, Median or Mode* of the feature and replace it with the missing values.


Mean (Download Speed) = 130

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Lite	99	70%
3	Fast+	167	10%
4	Fast+	N/A	80%
5	Lite	76	70%
6	Fast+	155	10%
7	Fast+	N/A	95%
8	Lite	76	77%
9	Fast+	180	95%

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Lite	99	70%
3	Fast+	167	10%
4	Fast+	130	80%
5	Lite	76	70%
6	Fast+	155	10%
7	Fast+	130	95%
8	Lite	76	77%
9	Fast+	180	95%

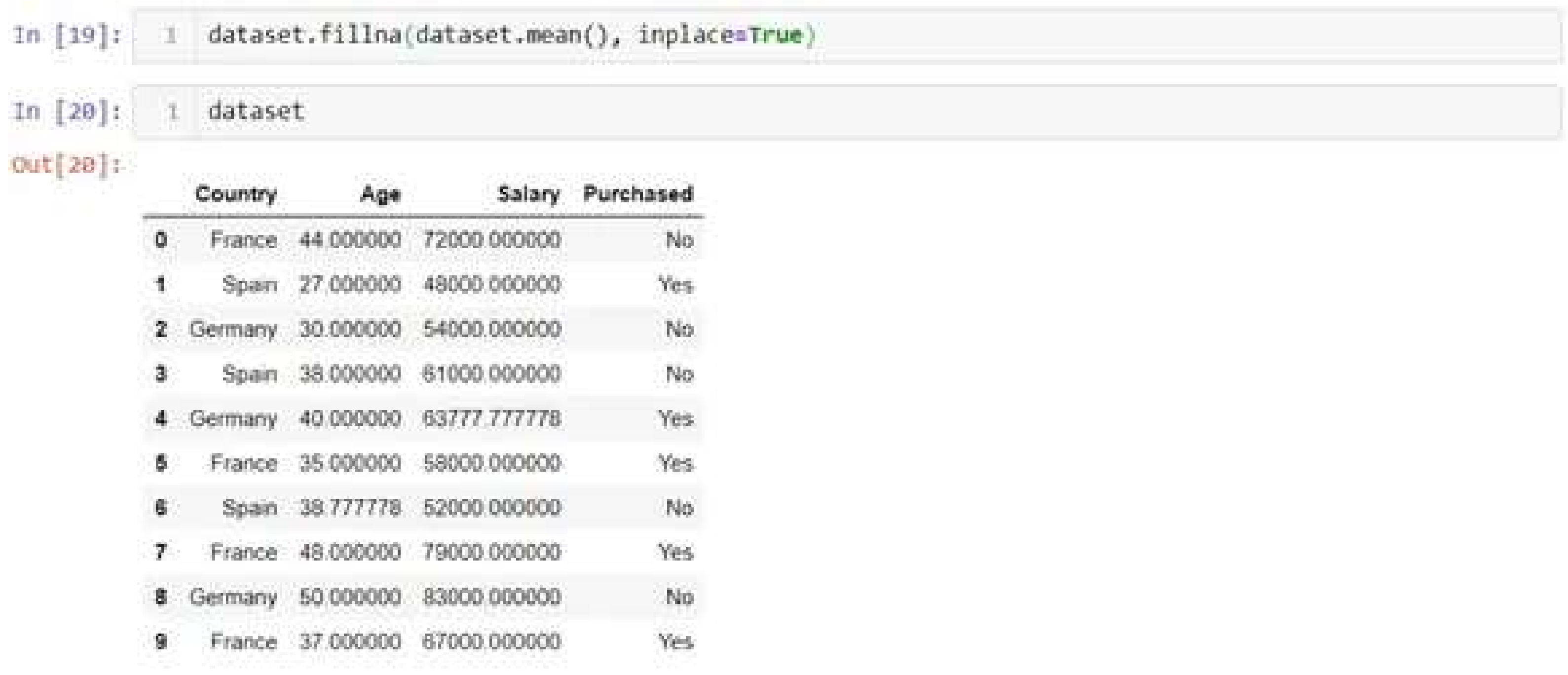

Median (Download Speed) = 155

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Lite	99	70%
3	Fast+	167	10%
4	Fast+	N/A	80%
5	Lite	76	70%
6	Fast+	155	10%
7	Fast+	N/A	95%
8	Lite	76	77%
9	Fast+	180	95%

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Lite	99	70%
3	Fast+	167	10%
4	Fast+	155	80%
5	Lite	76	70%
6	Fast+	155	10%
7	Fast+	155	95%
8	Lite	76	77%
9	Fast+	180	95%

Data Pre processing and Feature Engineering

Replacing with Mean



```
In [19]:  dataset.fillna(dataset.mean(), inplace=True)

In [20]:  dataset
Out[20]:
```

	Country	Age	Salary	Purchased
0	France	44.000000	72000.000000	No
1	Spain	27.000000	48000.000000	Yes
2	Germany	30.000000	54000.000000	No
3	Spain	38.000000	61000.000000	No
4	Germany	40.000000	63777.777778	Yes
5	France	35.000000	58000.000000	Yes
6	Spain	38.777778	52000.000000	No
7	France	48.000000	79000.000000	Yes
8	Germany	50.000000	83000.000000	No
9	France	37.000000	67000.000000	Yes

In the above line of code, it will affect the entire data-set and replaces every variable null values with their respective mean, and ‘*inplace =True*’ indicates to affect the changes to dataset

Data Pre processing and Feature Engineering

Categorical Imputation: When dealing with categorical columns, replacing missing values with the highest value in the column is a smart solution, or substitute missing values of each attribute by the most common value of the attribute

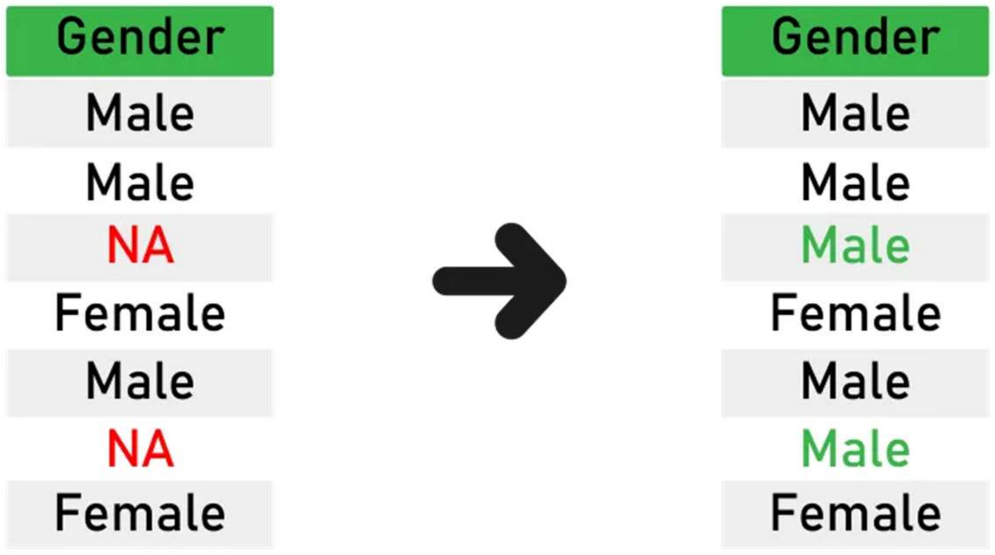
Eg.

#Max fill function for categorical columns

```
data['column_name'].fillna(data['column_name'].value_counts().idxmax(), inplace=True)
```

Data Pre processing and Feature Engineering

Frequent category imputation — or **mode imputation** — consists of replacing all occurrences of missing values (NA) within a variable with the mode, or the most frequent value.

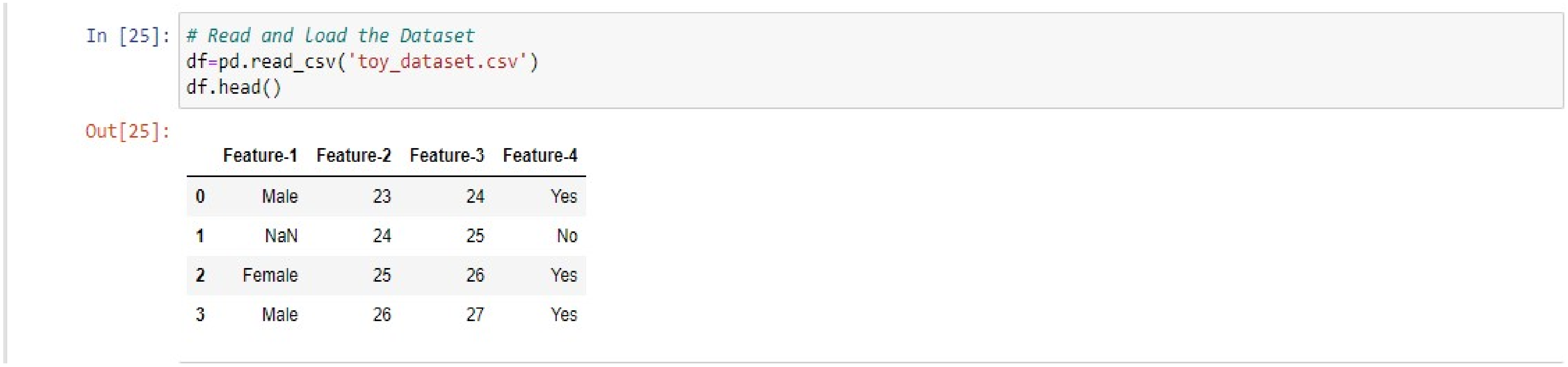


Gender
Male
Male
NA
Female
Male
NA
Female

Gender
Male
Male
Male
Female
Male
Male
Female

Frequent Category Imputation

Data Pre processing and Feature Engineering

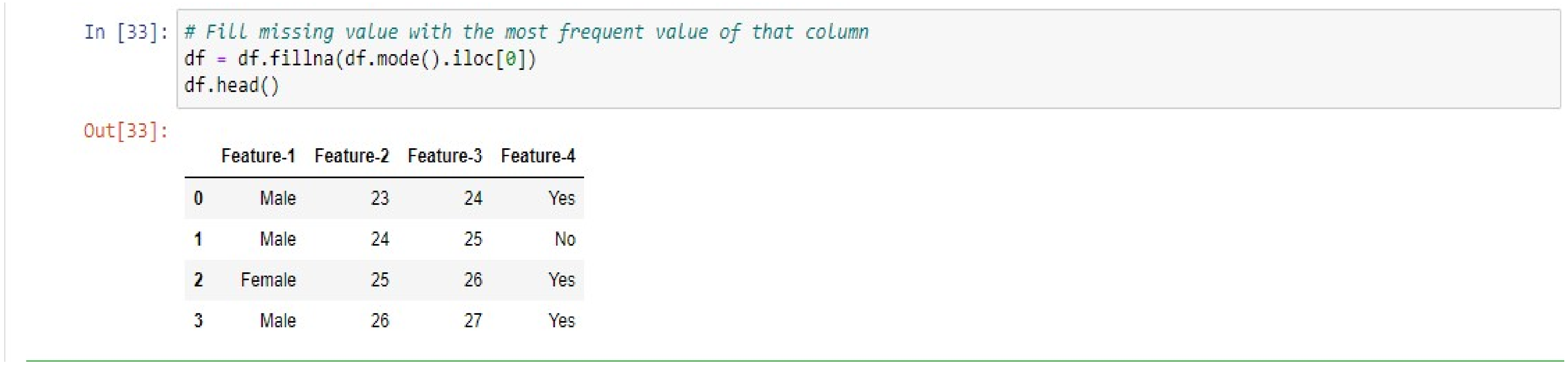


```
In [25]: # Read and Load the Dataset
         df=pd.read_csv('toy_dataset.csv')
         df.head()
```

Out[25]:

	Feature-1	Feature-2	Feature-3	Feature-4
0	Male	23	24	Yes
1	NaN	24	25	No
2	Female	25	26	Yes
3	Male	26	27	Yes

The 1st index is havimg null value and it can be replaced with categorical imputation



```
In [33]: # Fill missing value with the most frequent value of that column
         df = df.fillna(df.mode().iloc[0])
         df.head()
```

Out[33]:

	Feature-1	Feature-2	Feature-3	Feature-4
0	Male	23	24	Yes
1	Male	24	25	No
2	Female	25	26	Yes
3	Male	26	27	Yes

Data Pre processing and Feature Engineering

b) Handling Outliers

Outliers are those data points that are significantly different from the rest of the dataset. They are often abnormal observations that skew the data distribution, and arise due to inconsistent data entry, or erroneous observations.

Outlier handling is a technique for removing outliers from a dataset. This method can be used on a variety of scales to produce a more accurate data representation.

Data Pre processing and Feature Engineering

The various methods of handling outliers include:

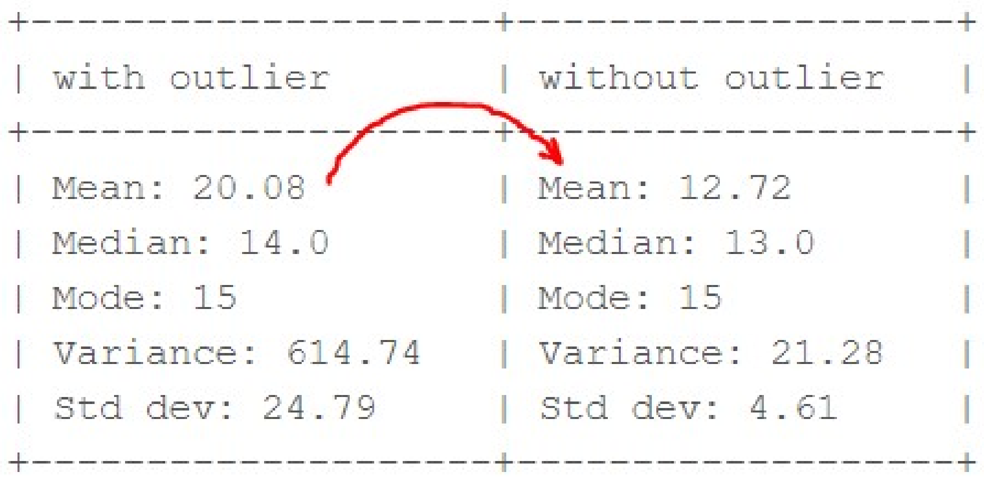
- **Removal**: Outlier-containing entries are deleted from the distribution. However, if there are outliers across numerous variables, this strategy may result in a big chunk of the datasheet being missed.

- **Replacing values**: Alternatively, the outliers could be handled as missing values and replaced with suitable imputation.

- **Capping**: Using an arbitrary value or a value from a variable distribution to replace the maximum and minimum values.

Data Pre processing and Feature Engineering

Example
Consider a small dataset, sample= [15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]. By looking at it, one can quickly say '101' is an outlier that is much larger than the other values.



```
+--------------------+--------------------+
| with outlier       | without outlier    |
+--------------------+--------------------+
| Mean: 20.08        | Mean: 12.72        |
| Median: 14.0       | Median: 13.0       |
| Mode: 15           | Mode: 15           |
| Variance: 614.74   | Variance: 21.28    |
| Std dev: 24.79     | Std dev: 4.61      |
+--------------------+--------------------+
```

From the above calculations, we can clearly say the Mean is more affected than the Median.

Data Pre processing and Feature Engineering

Example
Consider a small dataset, sample= [15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]. By looking at it, one can quickly say ‘101’ is an outlier that is much larger than the other values.

1. **Remove the outliers**:-In this technique, we remove the outliers from the dataset.

```
# Trimming for i in sample_outliers: a = np.delete(sample, np.where(sample==i)) print(a) # print(len(sample), len(a))
```

Old dataset -[15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]
The outlier ‘101’ is deleted and the rest of the data points are copied to another array ‘a’.
New dataset -[15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]

Data Pre processing and Feature Engineering

2. **Quantile Based Flooring and Capping**

In this technique, the outlier is capped at a certain value above the 90th percentile value or floored at a factor below the 10th percentile value. Python code to delete the outlier and copy the rest of the elements to another array.

```
# Computing 10th, 90th percentiles and replacing the outliers
tenth_percentile = np.percentile(sample, 10)
ninetieth_percentile = np.percentile(sample, 90)
# print(tenth_percentile, ninetieth_percentile)b =
np.where(sample<tenth_percentile, tenth_percentile, sample)
b = np.where(b>ninetieth_percentile, ninetieth_percentile, b)
# print("Sample:", sample)
print("New array:",b)
```

Data Pre processing and Feature Engineering

The above code outputs: **New array:** [15, 20.7, 18, 7.2, 13, 16, 11, 20.7, 7.2, 15, 10, 9]

The data points that are lesser than the 10th percentile are replaced with the 10th percentile value and the data points that are greater than the 90th percentile are replaced with 90th percentile value.

Data Pre processing and Feature Engineering

2. DataTransformation

Data transformation is the process of converting raw data into a a format or structure that would be more suitable for the model or algorithm and also data discovery in general.

1. Log Transform
2. Square Root Transform
3. Reciprocal Transformation
4. Box-Cox Transformation

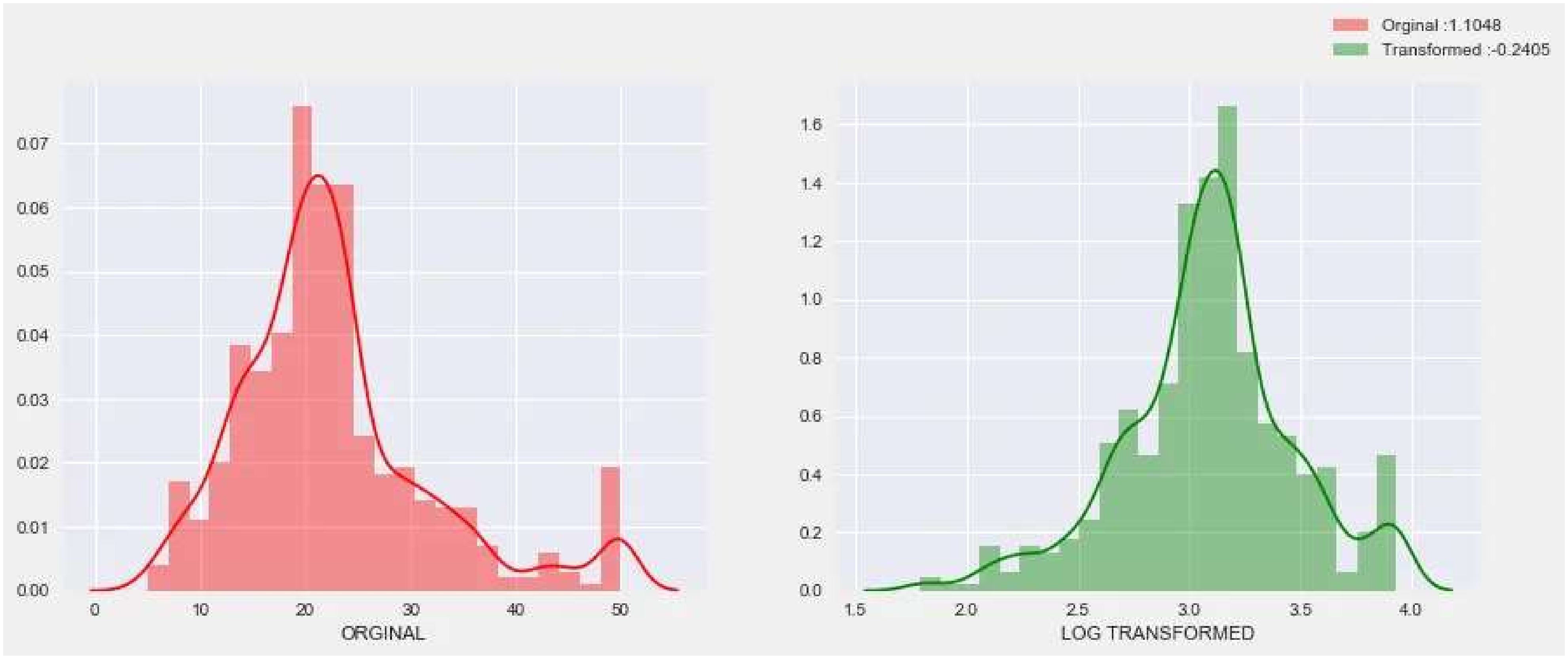
Data Pre processing and Feature Engineering

3. Log Transform

Log Transform is the most used technique among data scientists. It’s mostly used to turn a skewed distribution into a normal or less-skewed distribution. We take the log of the values in a column and utilise those values as the column in this transform. It is used to handle confusing data, and the data becomes more approximative to normal applications.

```
import numpy as np
log_target = np.log1p(df["Target"])
```

Data Pre processing and Feature Engineering



The above plot is the comparison of original and Log transformed data. Here we see the skewness is reduced in the transformed data. (best skew value should be nearly zero)

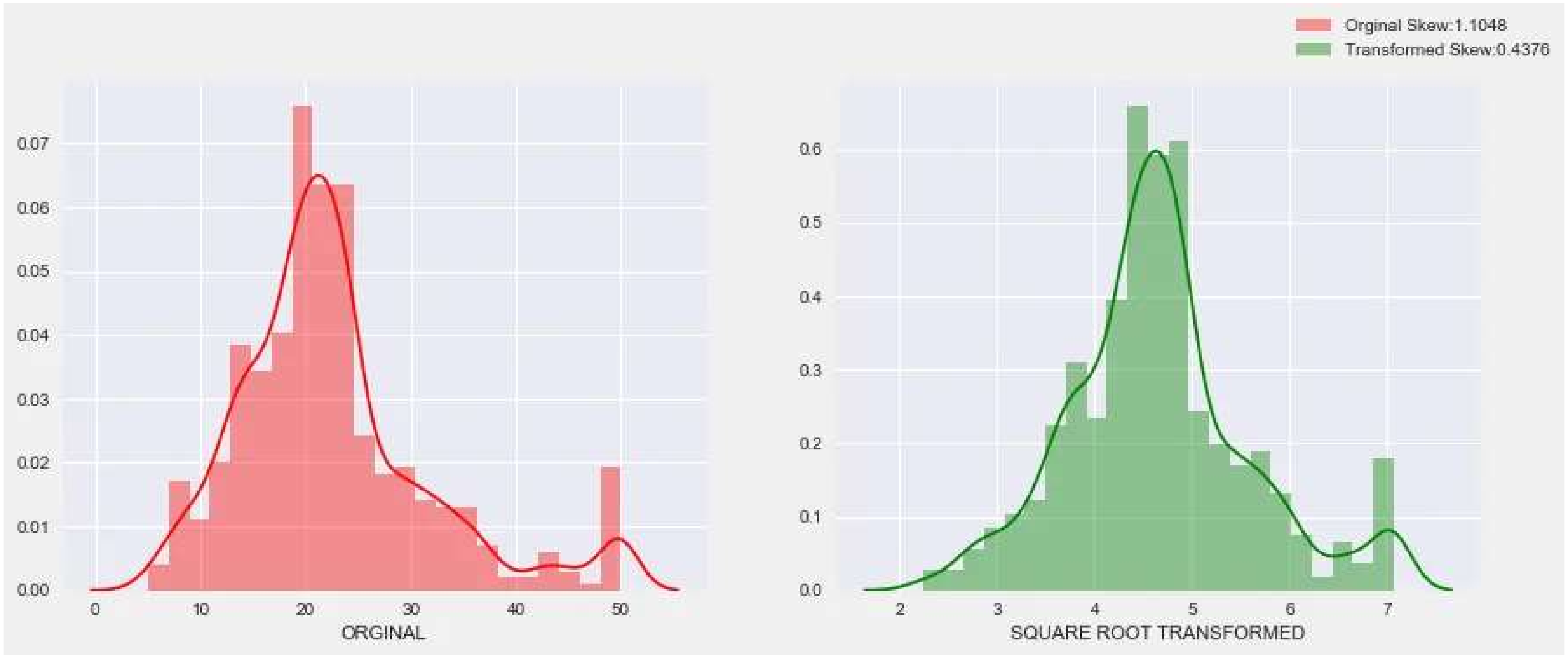
Data Pre processing and Feature Engineering

2. Square-Root Transformation

This transformation will give a moderate effect on distribution. The main advantage of square root transformation is, it can be applied to zero values. Here the x will replace by the square root(x). It is weaker than the Log Transformation.

```
sqrt_target = df["Target"]**(1/2)
```

Data Pre processing and Feature Engineering



Data Pre processing and Feature Engineering

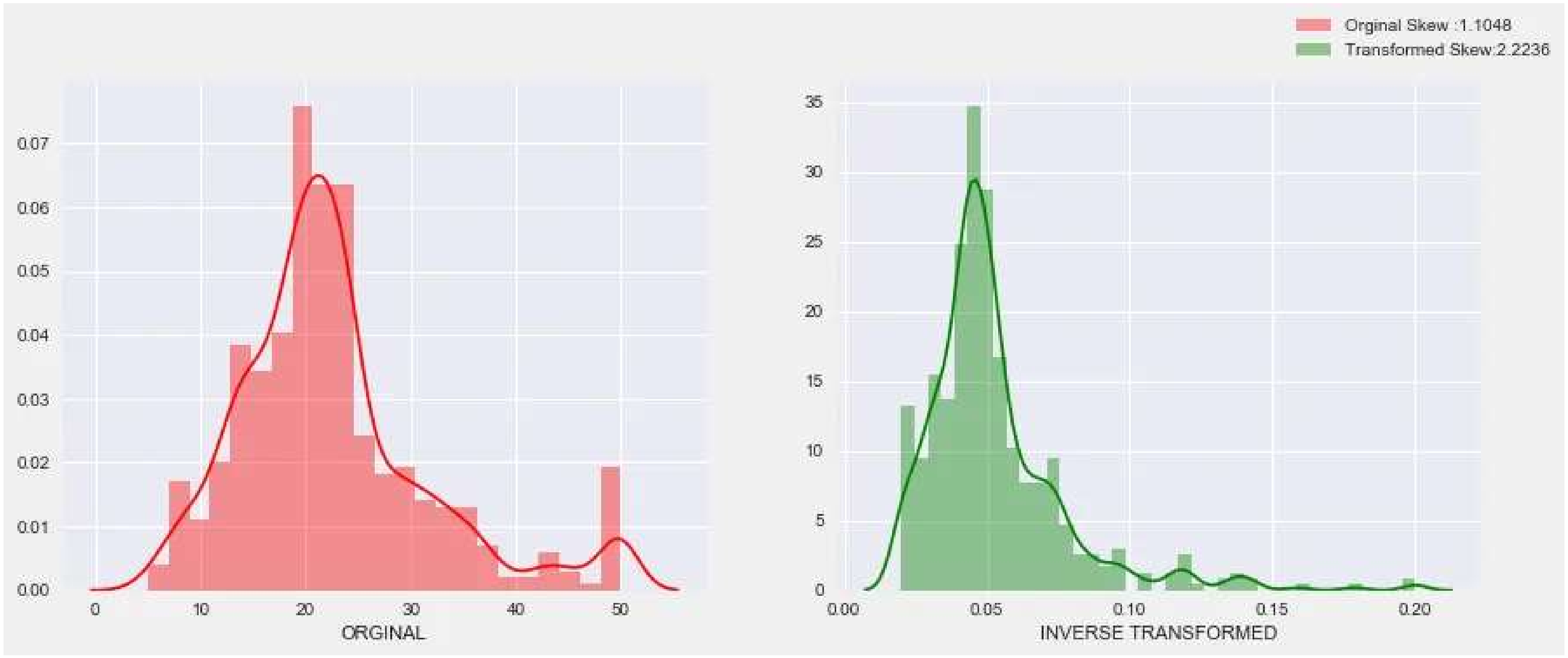
3. Reciprocal Transformation :

In this transformation, x will replace by the inverse of x (1/x).

The reciprocal transformation will give little effect on the shape of the distribution. This transformation can be only used for non-zero values.

```
reciprocal_target = 1/df["Target"]
```

Data Pre processing and Feature Engineering



The skewness for the transformed data is increased.

Data Pre processing and Feature Engineering

4. Box-Cox Transformation:

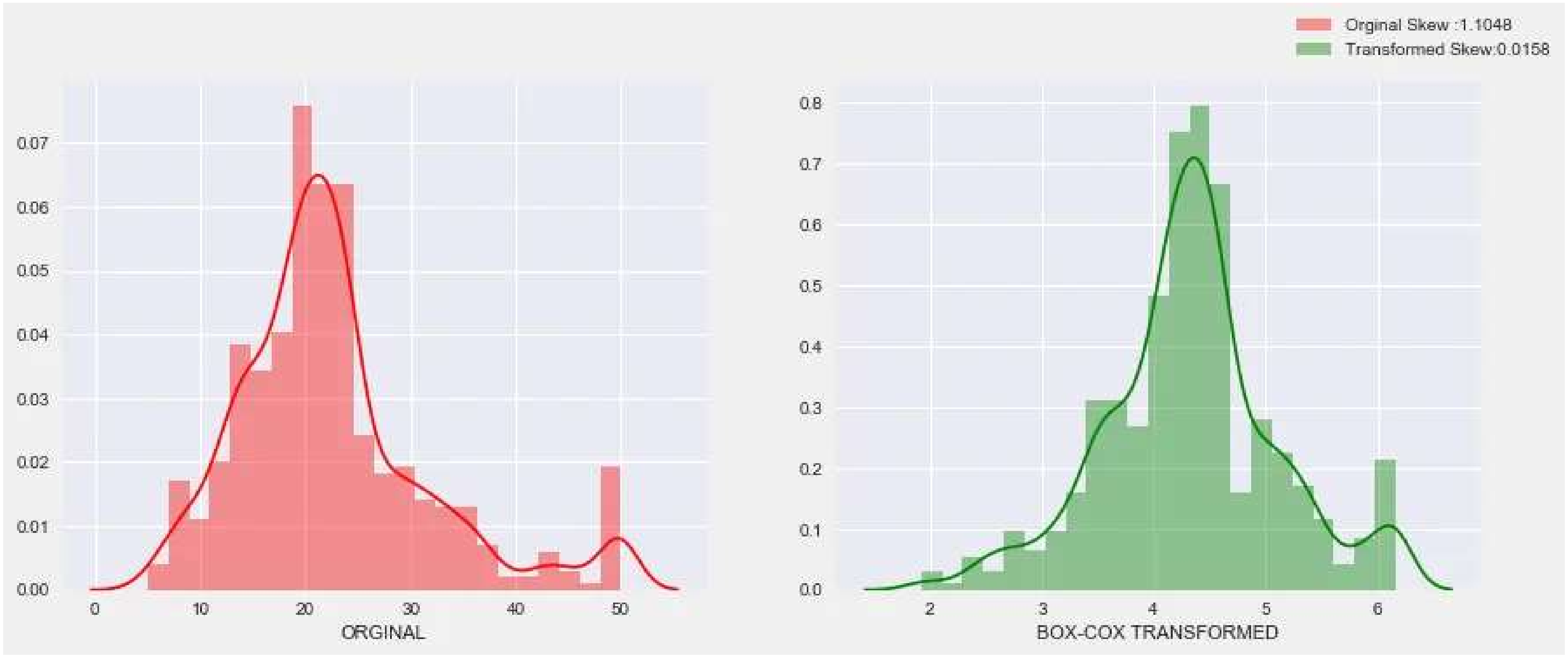
Box-cox transformation works pretty well for many data natures. The below image is the mathematical formula for Box-cox transformation.

$$y(\lambda) = \begin{cases} \frac{y^\lambda - 1}{\lambda}, & \text{if } \lambda \neq 0; \\ \log y, & \text{if } \lambda = 0. \end{cases}$$

All the values of lambda vary from -5 to 5 are considered and the best value for the data is selected. The “Best” value is one that results in the best skewness of the distribution.

```
from scipy.stats import boxcox
bcx_target, lam = boxcox(df["Target"])
#lam is the best lambda for the distribution
```

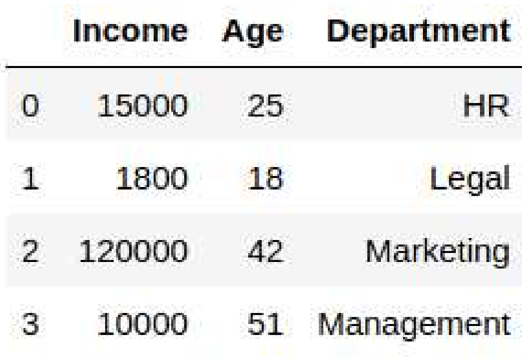
Data Pre processing and Feature Engineering



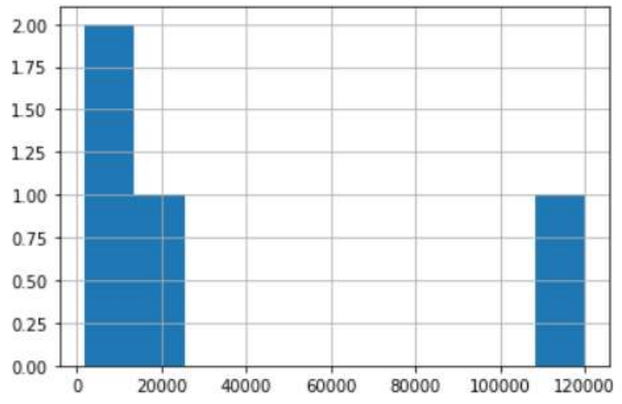
Here, we noticed that the Box-cox function reduced the skewness and it is almost equal to zero. Worked well ; For this transformation, values strictly to be positive.

Data Pre processing and Feature Engineering

Consider an example for Log Transfrom with below dataset with features Income,Age and Management

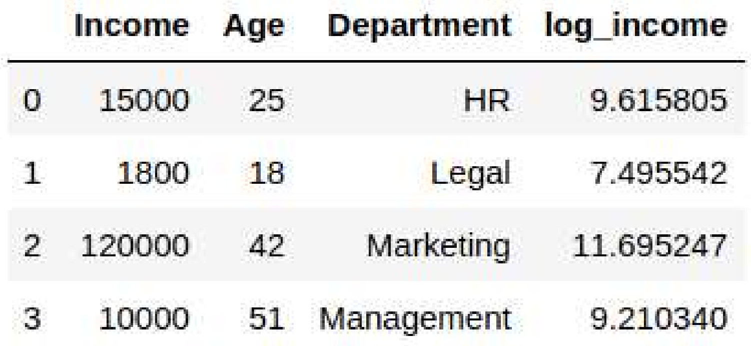


	Income	Age	Department
0	15000	25	HR
1	1800	18	Legal
2	120000	42	Marketing
3	10000	51	Management

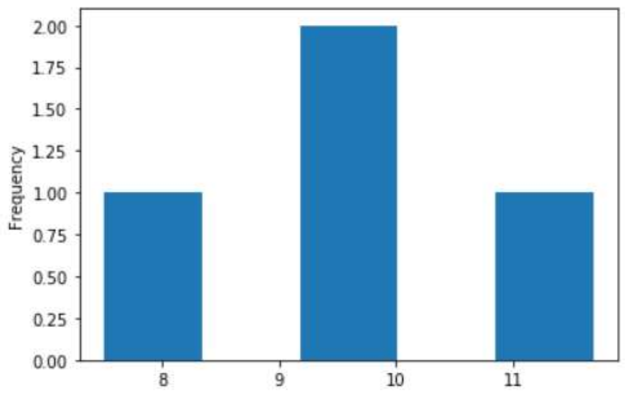


Apply log on income column:
df['log_income'] = np.log(df['Income'])
# We created a new column to store the log values

Data Pre processing and Feature Engineering



	Income	Age	Department	log_income
0	15000	25	HR	9.615805
1	1800	18	Legal	7.495542
2	120000	42	Marketing	11.695247
3	10000	51	Management	9.210340



While the Income column had extreme values ranging from 1800 to 1,20,000 – the log values are now ranging from approximately 7.5 to 11.7

Thus, the log operation had a dual role:

•Reducing the impact of too-low values
•Reducing the impact of too-high values.

Data Pre processing and Feature Engineering

<u>4. Data Reduction</u>

Data reduction is a technique used to reduce the size of a dataset while still preserving the most important information. This can be beneficial in situations where the dataset is too large to be processed efficiently, or where the dataset contains a large amount of irrelevant or redundant information.The methods are:-

- **Data Sampling**
- **Dimensionality Reduction**
- **Data Compression**
- **Data Discretization**
- **Feature Selection**

Data Pre processing and Feature Engineering

Data Sampling: This technique involves selecting a subset of the data to work with, rather than using the entire dataset. This can be useful for reducing the size of a dataset while still preserving the overall trends and patterns in the data.

Dimensionality Reduction: This technique involves reducing the number of features in the dataset, either by removing features that are not relevant or by combining multiple features into a single feature.

Data Pre processing and Feature Engineering

Data Compression: This technique involves using techniques such as lossy or lossless compression to reduce the size of a dataset.

Data Discretization: This technique involves converting continuous data into discrete data by partitioning the range of possible values into intervals or bins.

Feature Selection: This technique involves selecting a subset of features from the dataset that are most relevant to the task at hand.

Data Pre processing and Feature Engineering

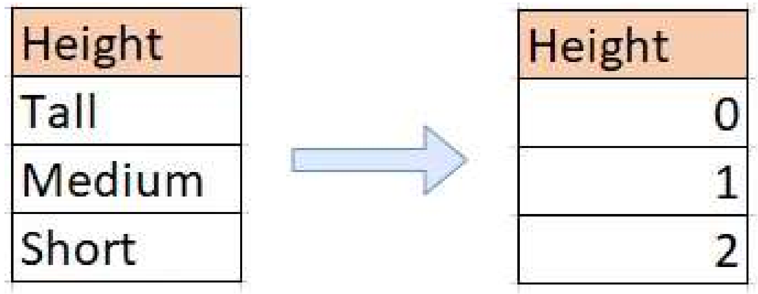
4. Categorical Encoding

The process of encoding categorical data into numerical data is called "categorical encoding." It involves transforming categorical variables into a numerical format suitable for machine learning models.

1. ***Label Encoding or Ordinal Encoding***
2. ***One-hot encoding***
3. ***Effect Encoding***
4. ***Binary Encoding***
5. ***Base-N Encoding***
6. ***Hash Encoding***
7. ***Target Encoding***

1. Label Encoding or Ordinal Encoding:

This encoding technique assigns a unique integer value to each category of the categorical variable. It is used when the categorical variable has an inherent order or hierarchy. For example, encoding T-shirt sizes as 0 for Small, 1 for Medium, and 2 for Large.



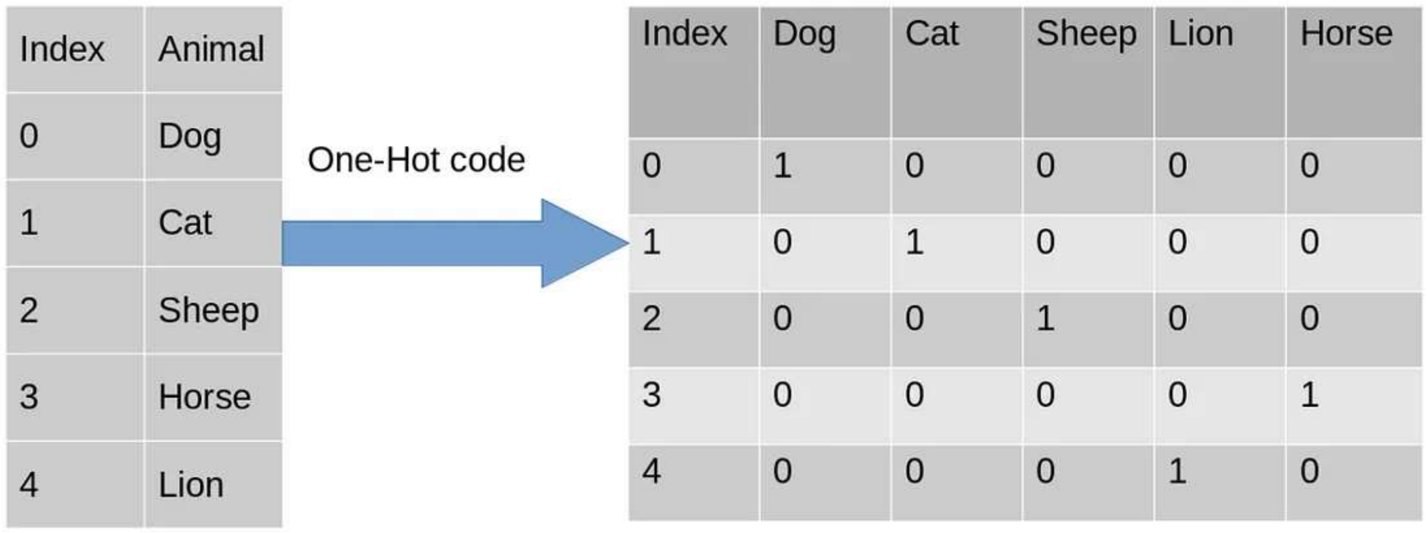
Height
Tall
Medium
Short

Height
0
1
2

Data Pre processing and Feature Engineering

2. One-hot encoding:

One-Hot Encoding creates new binary variables (dummy variables) for each category of the categorical variable. Each category is represented by a binary feature, where 1 indicates the presence of that category and 0 indicates the absence. For example, encoding colors as Red [1, 0, 0], Green [0, 1, 0], Blue [0, 0, 1].



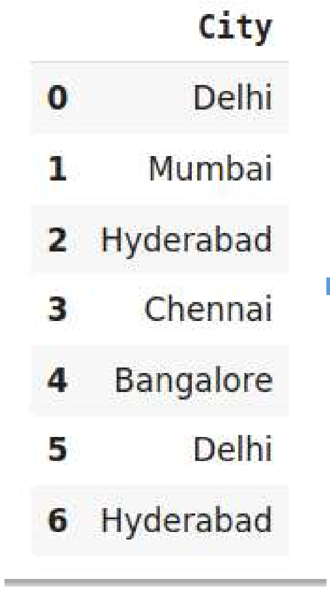
Index	Animal
0	Dog
1	Cat
2	Sheep
3	Horse
4	Lion

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

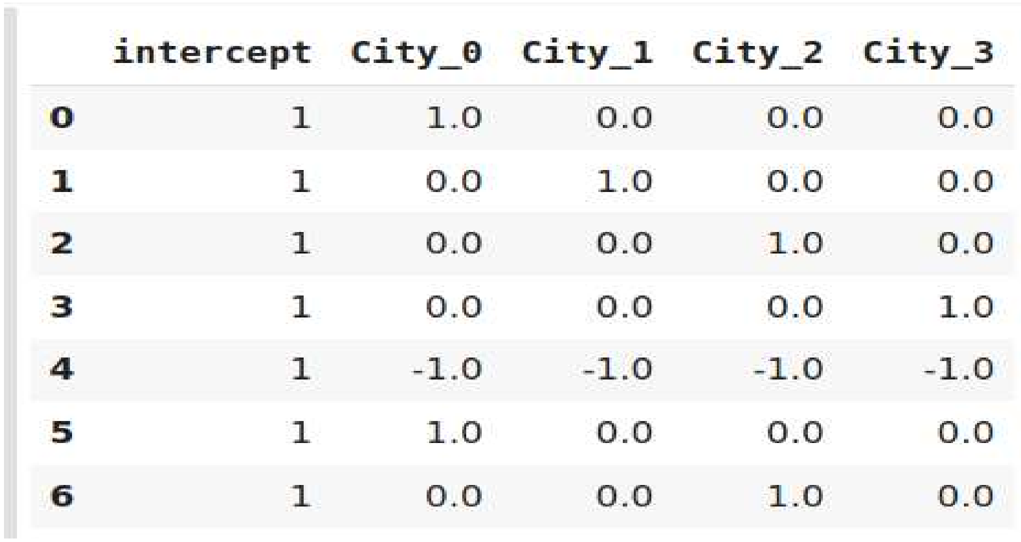
Data Pre processing and Feature Engineering

3. Effect Encoding:

Effect encoding (also known as Helmert encoding) is similar to one-hot encoding, but with a slight difference. In effect encoding, each category is represented by a series of values (-1, 0, 1), indicating the contrast between that category and the average of the subsequent categories. It is commonly used in regression models.



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

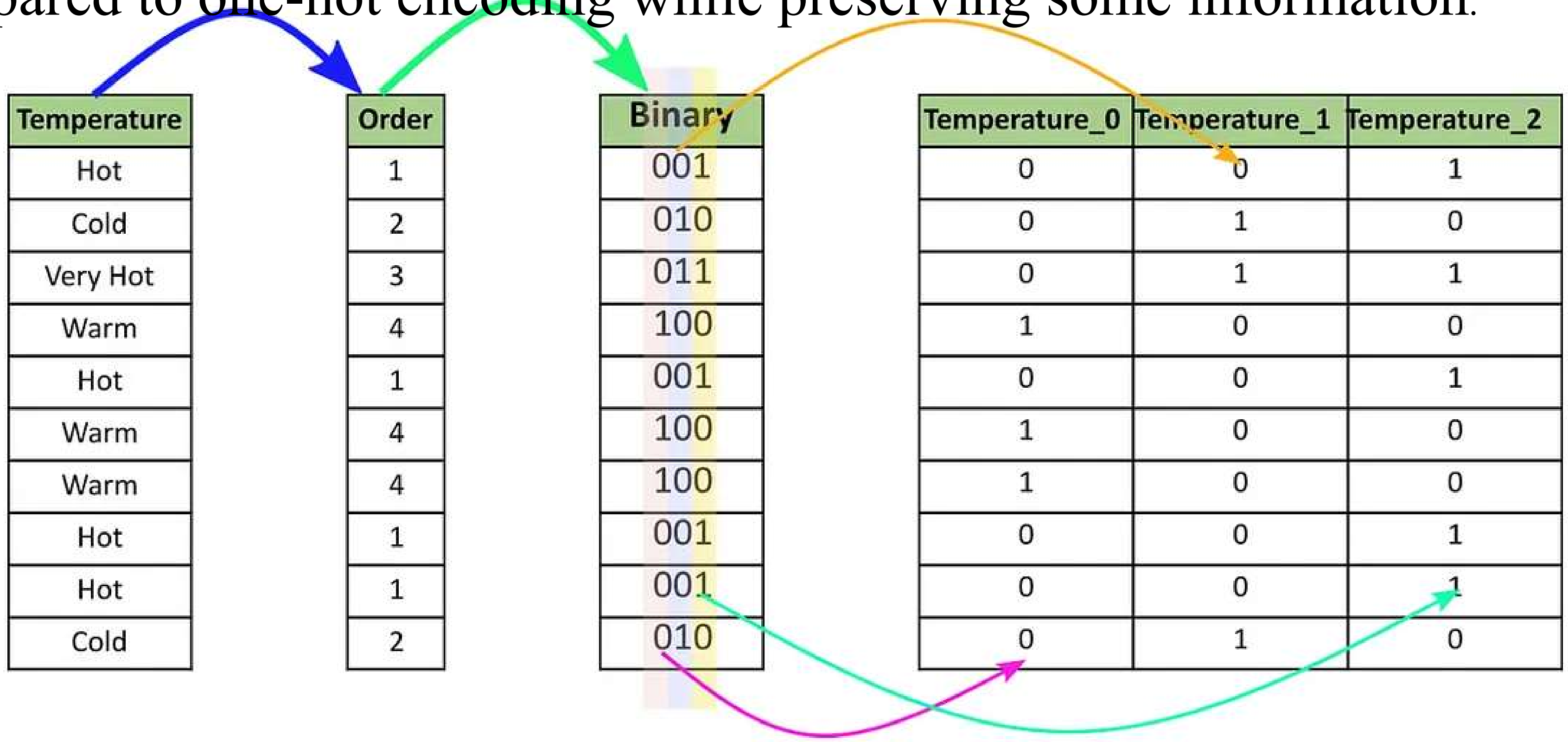


	intercept	City_0	City_1	City_2	City_3
0	1	1.0	0.0	0.0	0.0
1	1	0.0	1.0	0.0	0.0
2	1	0.0	0.0	1.0	0.0
3	1	0.0	0.0	0.0	1.0
4	1	-1.0	-1.0	-1.0	-1.0
5	1	1.0	0.0	0.0	0.0
6	1	0.0	0.0	1.0	0.0

Data Pre processing and Feature Engineering

4. **Binary Encoding**:

Binary encoding combines the advantages of hashing and one-hot encoding. It represents each category with binary digits, where each digit corresponds to a new binary variable. This encoding reduces the dimensionality compared to one-hot encoding while preserving some information.



Temperature
Hot
Cold
Very Hot
Warm
Hot
Warm
Warm
Hot
Hot
Cold

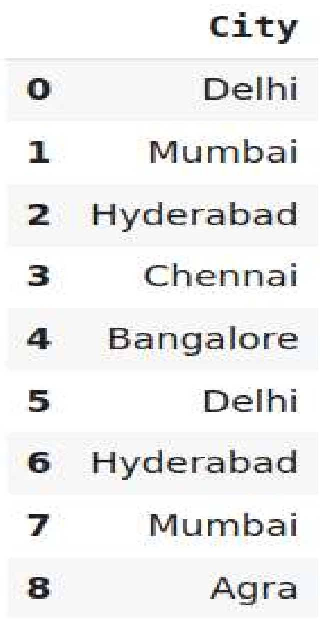
Order
1
2
3
4
1
4
4
1
1
2

Binary
001
010
011
100
001
100
100
001
001
010

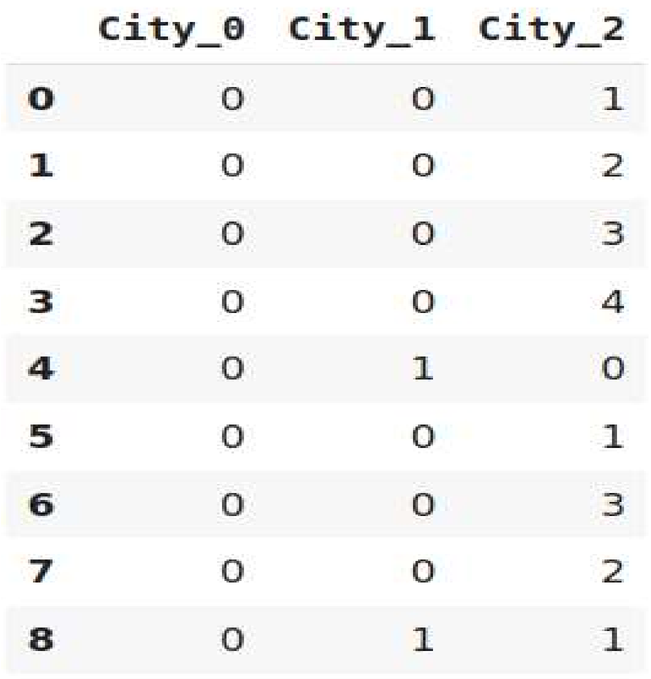
Temperature_0	Temperature_1	Temperature_2
0	0	1
0	1	0
0	1	1
1	0	0
0	0	1
1	0	0
1	0	0
0	0	1
0	0	1
0	1	0

Data Pre processing and Feature Engineering

5. **Base-N Encoding**:
Base-N encoding converts each category into a numerical form based on a specified base or radix. For example, in base 2 (binary), each category is encoded using its binary representation. Changing the base can lead to different numeric representations.



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

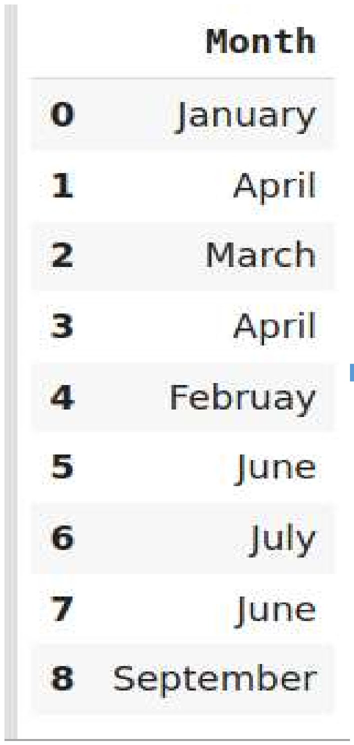
Here used base 5

Data Pre processing and Feature Engineering

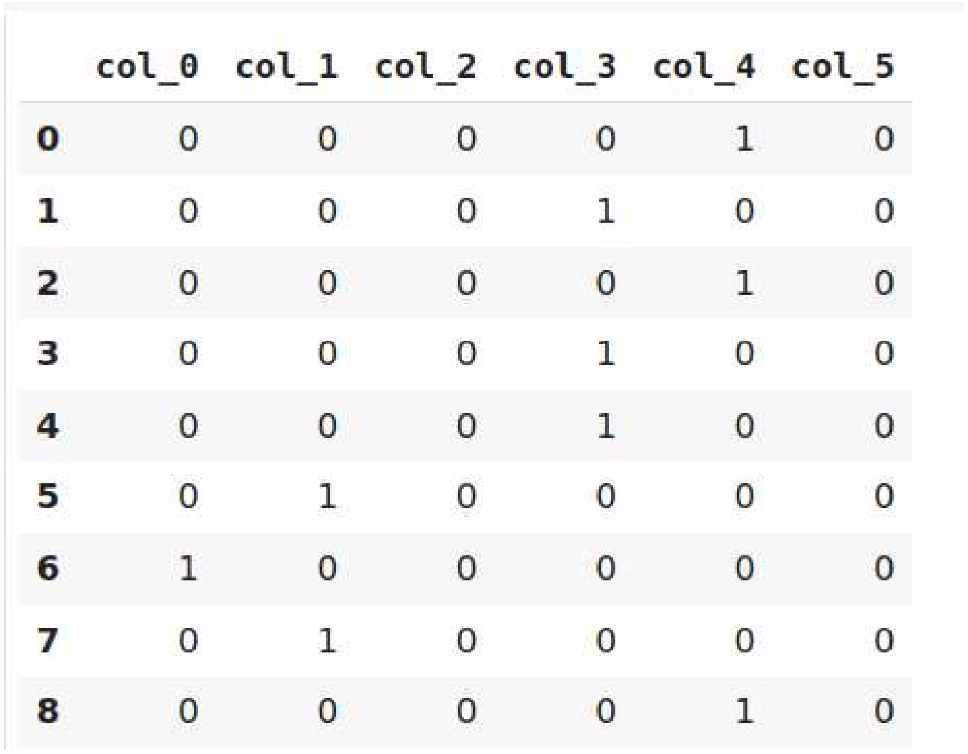
6. **Hash Encoding**:

Hash encoding converts categories into a fixed number of new variables using hashing techniques. The number of new variables is predefined, and hashing allows mapping the categories to these variables. However, there is a possibility of information loss due to collisions in the hashing process.

.



	Month
0	January
1	April
2	March
3	April
4	Februay
5	June
6	July
7	June
8	September

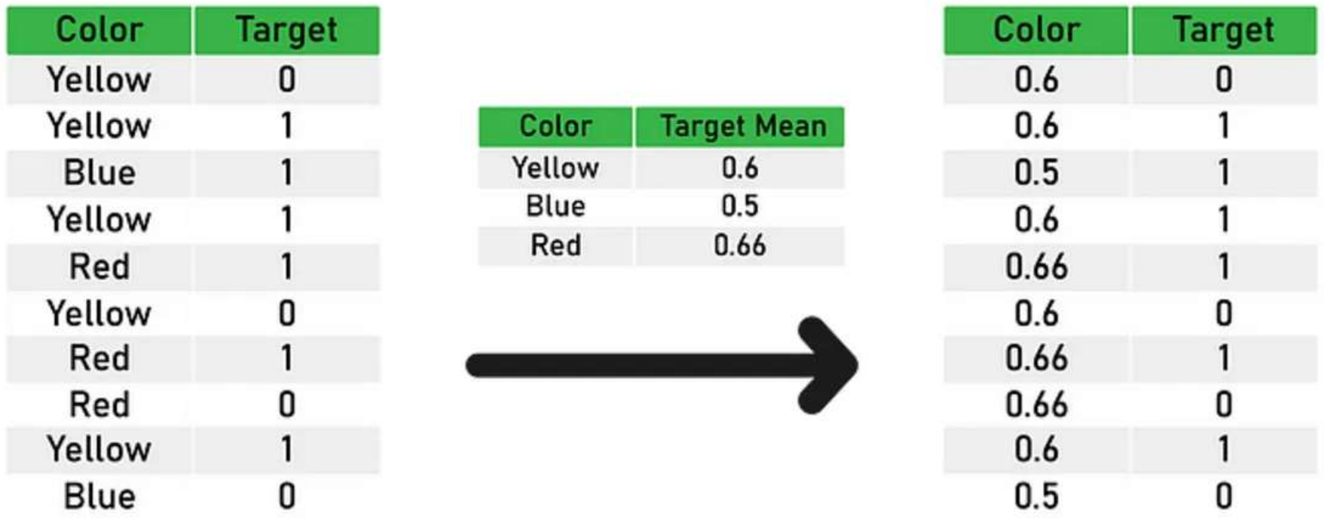


	col_0	col_1	col_2	col_3	col_4	col_5
0	0	0	0	0	1	0
1	0	0	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	1	0	0
4	0	0	0	1	0	0
5	0	1	0	0	0	0
6	1	0	0	0	0	0
7	0	1	0	0	0	0
8	0	0	0	0	1	0

Data Pre processing and Feature Engineering

7. Target Encoding:

Target encoding (also known as mean encoding) uses the target variable to encode the categorical values. It replaces each category with the mean value of the target variable for that category. Target encoding is useful when the categorical variable is believed to have a predictive relationship with the target variable.



Color	Target
Yellow	0
Yellow	1
Blue	1
Yellow	1
Red	1
Yellow	0
Red	1
Red	0
Yellow	1
Blue	0

Color	Target Mean
Yellow	0.6
Blue	0.5
Red	0.66

Color	Target
0.6	0
0.6	1
0.5	1
0.6	1
0.66	1
0.6	0
0.66	1
0.66	0
0.6	1
0.5	0

Data Pre processing and Feature Engineering

<u>5. Feature Scaling</u>

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. Scaling guarantees that all features are on a comparable scale and have comparable ranges. There are two common ways for scaling :

- Normalization
- Standardization

Data Pre processing and Feature Engineering

- **Normalization.**

The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.All values are scaled in a specified range between 0 and 1 via normalisation (or min-max normalisation).

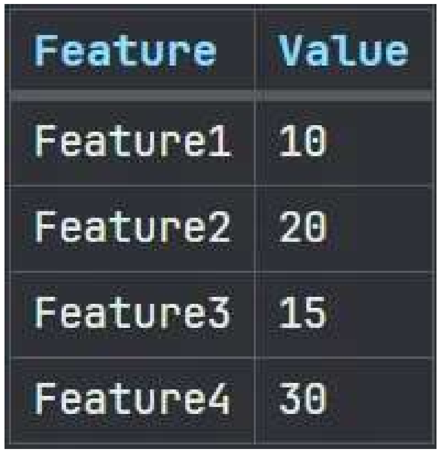
Formula:

X_scaled = (X — X_min) / (X_max — X_min),

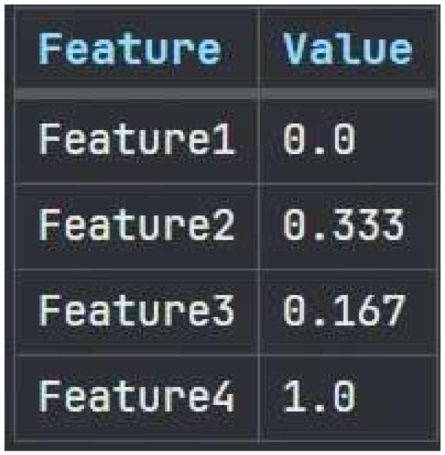
where X is the original feature, X_scaled is the scaled feature,

X_min is the minimum value of X, and X_max is the maximum value of X.

Data Pre processing and Feature Engineering



Feature	Value
Feature1	10
Feature2	20
Feature3	15
Feature4	30



Feature	Value
Feature1	0.0
Feature2	0.333
Feature3	0.167
Feature4	1.0

In this example, min-max scaling rescales the values of each feature to a specific range (typically between 0 and 1). The minimum value in the original dataset becomes 0, and the maximum value becomes 1. The other values are scaled proportionally within this range.

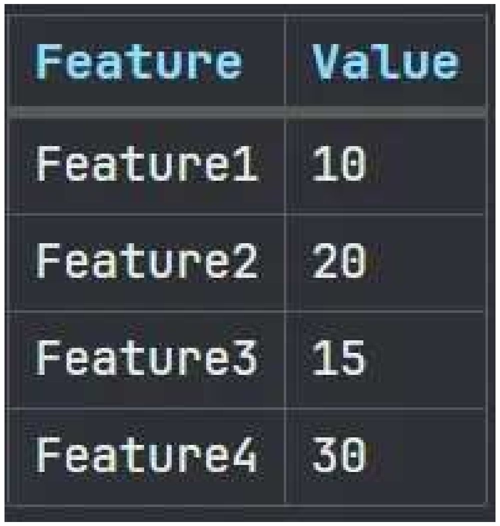
Data Pre processing and Feature Engineering

- **Standardization**:

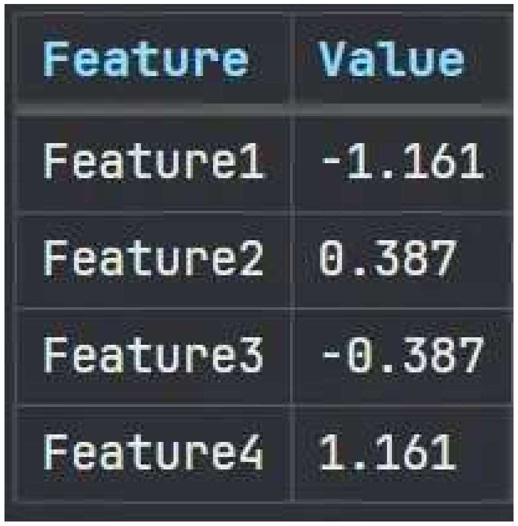
Standardization (also known as z-score normalisation) is the process of scaling values while accounting for standard deviation. The values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero, and the resultant distribution has a unit standard

Formula:
X_scaled = (X — X_mean) / X_std,
where X is the original feature, X_scaled is the scaled feature, X_mean is the mean of X, and X_std is the standard deviation of X.

Data Pre processing and Feature Engineering



Feature	Value
Feature1	10
Feature2	20
Feature3	15
Feature4	30



Feature	Value
Feature1	-1.161
Feature2	0.387
Feature3	-0.387
Feature4	1.161

In this example, standardization (Z-score scaling) transforms the values of each feature to have a mean of 0 and a standard deviation of 1. The original values are shifted and rescaled according to the mean and standard deviation of the feature.

References

1. Zheng, Alice, and Amanda Casari. Feature engineering for machine learning: principles and techniques for data scientists. " O'Reilly Media, Inc.", 2018.
2. Rawat, Tara, and Vineeta Khemchandani. "Feature engineering (FE) tools and techniques for better classification performance." International Journal of Innovations in Engineering and Technology 8.2 (2017): 169-179.
3. Gada, Mihir, et al. "Automated feature engineering and hyperparameter optimization for machine learning." *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*. Vol. 1. IEEE, 2021.
4. Hall, Ernest L., et al. "A survey of preprocessing and feature extraction techniques for radiographic images." *IEEE Transactions on Computers* 100.9 (1971): 1032-1044.
5. Liu, Huan, and Hiroshi Motoda, eds. *Feature extraction, construction and selection: A data mining perspective*. Vol. 453. Springer Science & Business Media, 1998.
6. https://medium.com/@denizgunay/feature-engineering-data-preprocessing
7. https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/
8. https://hpi.de/fileadmin/user_upload/fachgebiete/boettinger/documents/Kurse_WS_1920/Data_Management_for_Digital_Health/200116_Data_Preprocessing.pdf

THANKS