Experiment 5

Title: Apply Preprocessing techniques on dataset using filters: NumericToNominal, StringToNominal, NominalToBinary, Normalize. Also do the result analysis before and after preprocessing.

1. NumericToNominal filter: The NumericToNominal filter is a unsupervised attribute filter used to convert numeric attributes into nominal (categorical) attributes without binning or altering the original values. Unlike the "Discretize" filter, which divides numeric values into intervals, NumericToNominal simply relabels each distinct numeric value as a nominal category. This is useful when numeric values actually represent categorical labels rather than continuous measurements (e.g., 0 = No, 1 = Yes).

Choose Dataset: weather.numeric

Fig 5.1

Analysis: Some numeric values represent categories, not quantities (like outlook, windy, play). Algorithms may treat these categorical codes as continuous values, leading to incorrect model assumptions. Models such as decision trees or Naive Bayes expect nominal labels, especially for classification tasks like predicting play.

Before applying the "NumericToNominal filter"

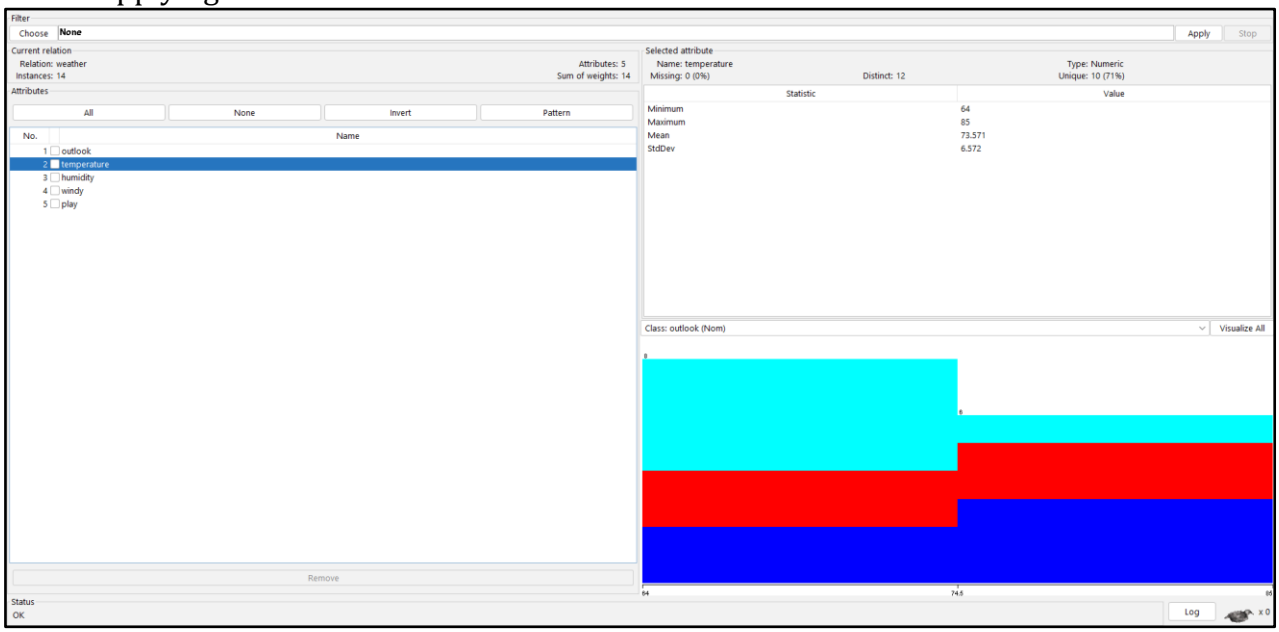

Fig 5.2

Description: Before applying the NumericToNominal filter on the temperature attribute, it is treated as a continuous numeric value. Algorithms interpret it based on mathematical relationships, considering differences and averages, rather than as distinct categories or ranges.

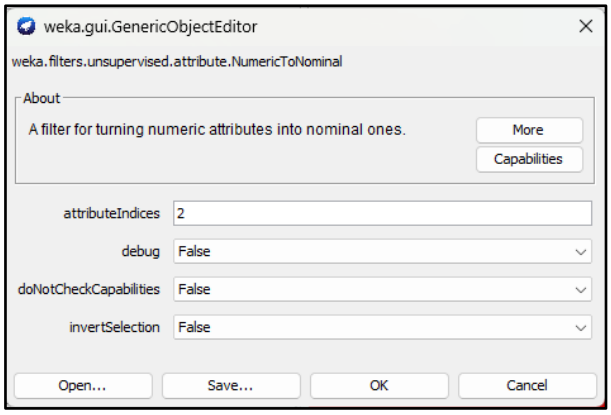

Fig 5.3

Description: Filter Section click on choose button> Filter> Unsupervised> Select NumericToNominal filter.

After applying the "NumericToNominal filter"

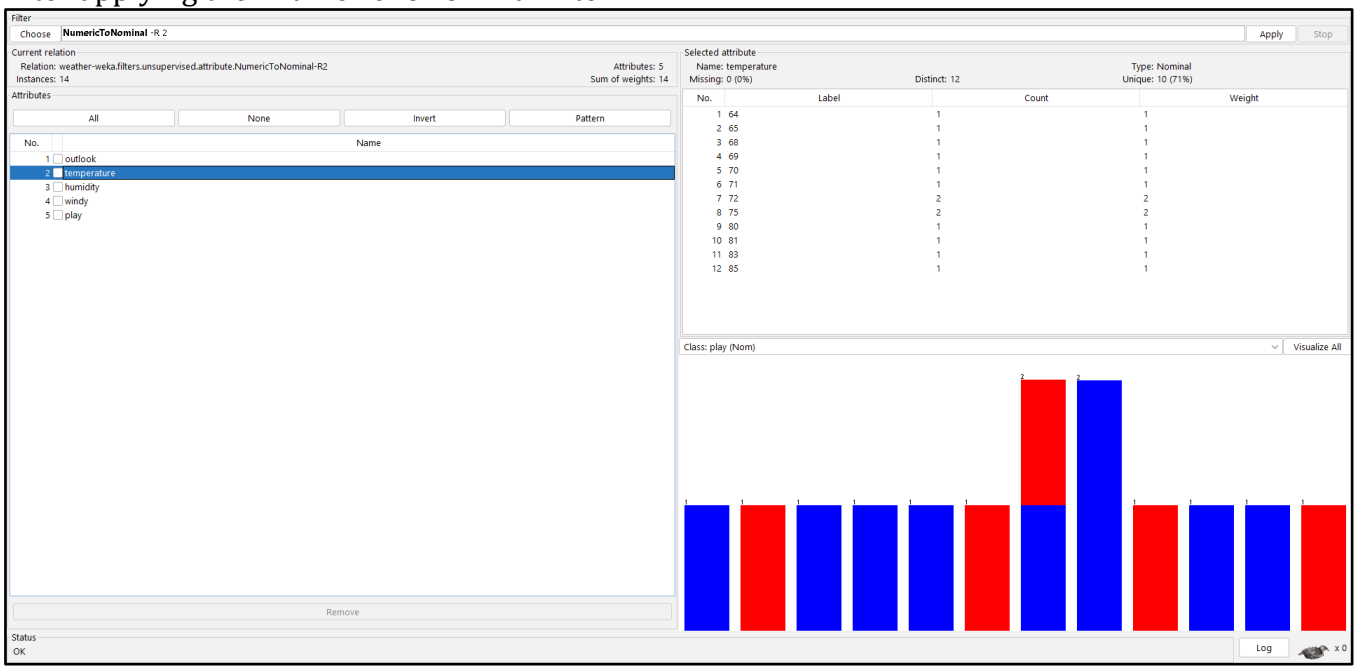

Fig 5.4

Description: After applying the NumericToNominal filter on the temperature attribute, the continuous numeric temperature values are converted into discrete categorical labels. This transformation may reduce precision but allows algorithms that handle only nominal data to process temperature as a set of distinct categories.

2. StringToNominal filter: The StringToNominal filter is a unsupervised attribute filter used to convert string attributes into nominal (categorical) attributes. This is useful because many machine learning algorithms in Weka cannot handle raw string-type attributes. By converting them to nominal format, the values become valid categorical inputs that can be processed during model training.

Choose Dataset: student

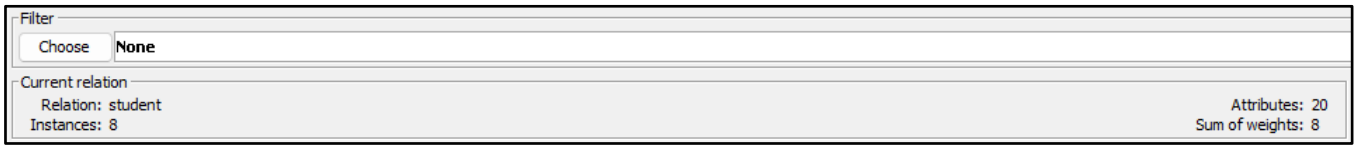

Fig 5.5

Analysis: StringToNominal is essential for attributes like Email_ID, Guide_Name, Branch, Hobby, State, City, and Gender to convert them into usable categorical formats. NumericToNominal can be applied to attributes such as Er_no, Mobile_no, and possibly Backlogs, if they represent categories rather than continuous values.Normalize should be applied to continuous numeric fields like CGPA to scale them uniformly.

Before applying the “StringToNominal filter”

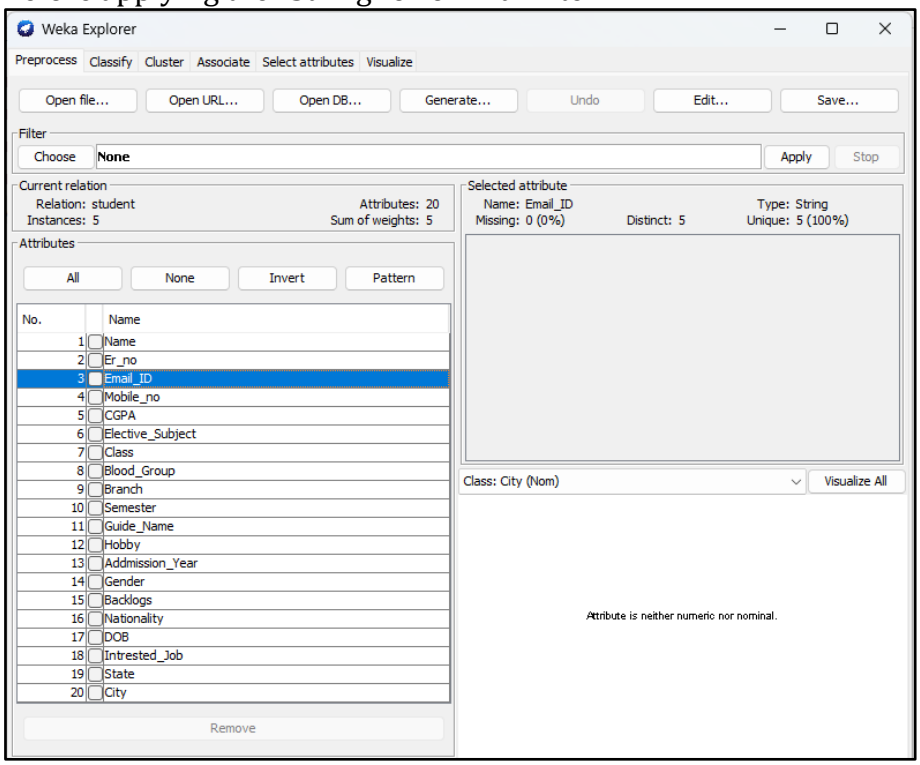

Fig 5.6

Description: The Email_ID attribute is stored as a string, and machine learning algorithms in Weka cannot process string attributes directly. As a result, it is either ignored or causes errors during model training.

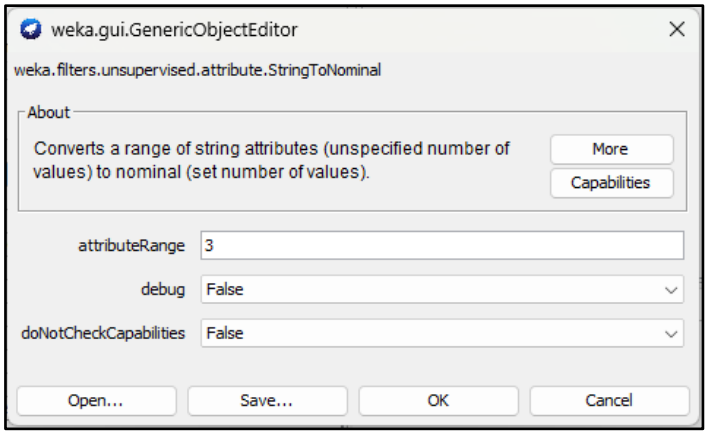

Fig 5.7

Filter Section click on choose button> Filter> Unsupervised> Select StringToNominal filter.

After applying the "StringToNominal filter"

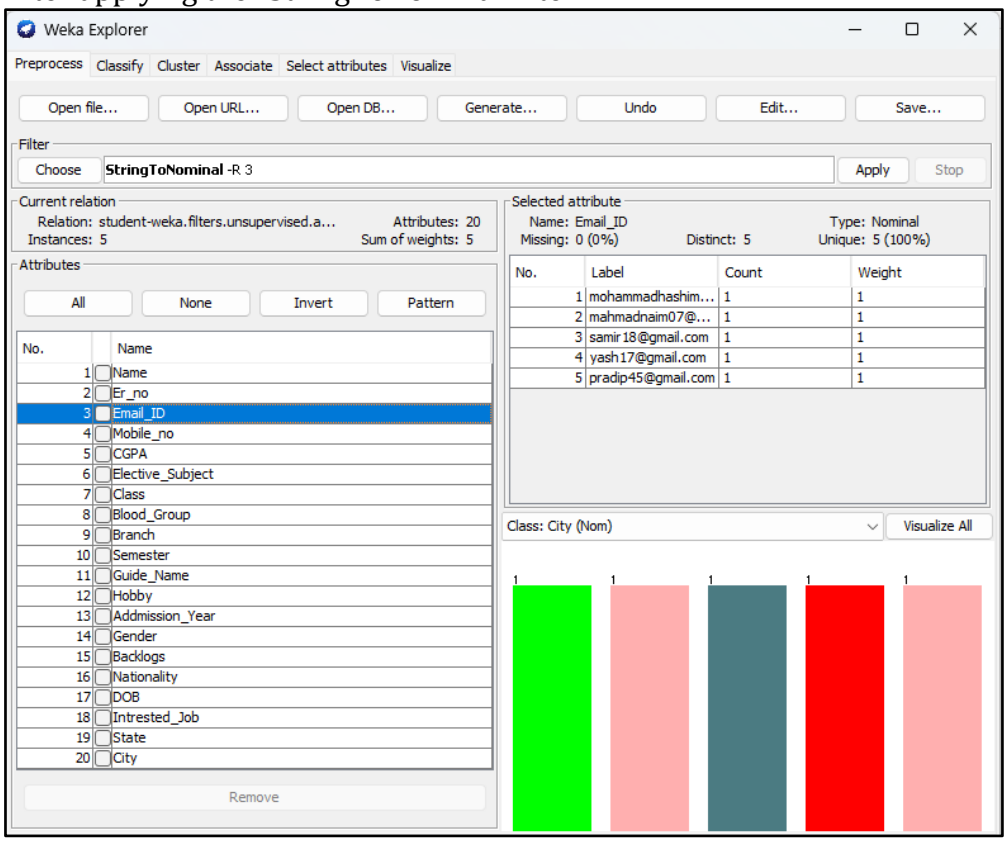

Fig 5.8

Description: The Email_ID attribute is converted into nominal (categorical) values, allowing it to be included in the training process. Each unique email is now treated as a distinct category, making it usable for classification or clustering tasks if relevant.

3. NominalToBinary filter: The NominalToBinary filter is a unsupervised attribute filter that converts nominal (categorical) attributes into binary numeric attributes using one-hot encoding. Each category of a nominal attribute is transformed into a separate binary (0 or 1) attribute, where:

- 1 indicates the presence of the category,
- 0 indicates its absence.

This transformation allows algorithms that require numeric input (e.g., linear regression, neural networks) to handle categorical data effectively.

Choose Dataset: weather.nominal

Fig 5.9

Analysis: outlook – nominal (e.g., sunny, overcast, rainy), temperature – nominal (e.g., hot, mild, cool), humidity – nominal (e.g., high, normal), windy – nominal (e.g., TRUE, FALSE), play – nominal (e.g., yes, no). All attributes are nominal (categorical), suitable for classification tasks. No numeric or string types are present. Ideal for decision trees, Naive Bayes, rule-based classifiers. The class label is play, which we aim to predict based on weather conditions.

Before applying the "NominalToBinary filter"

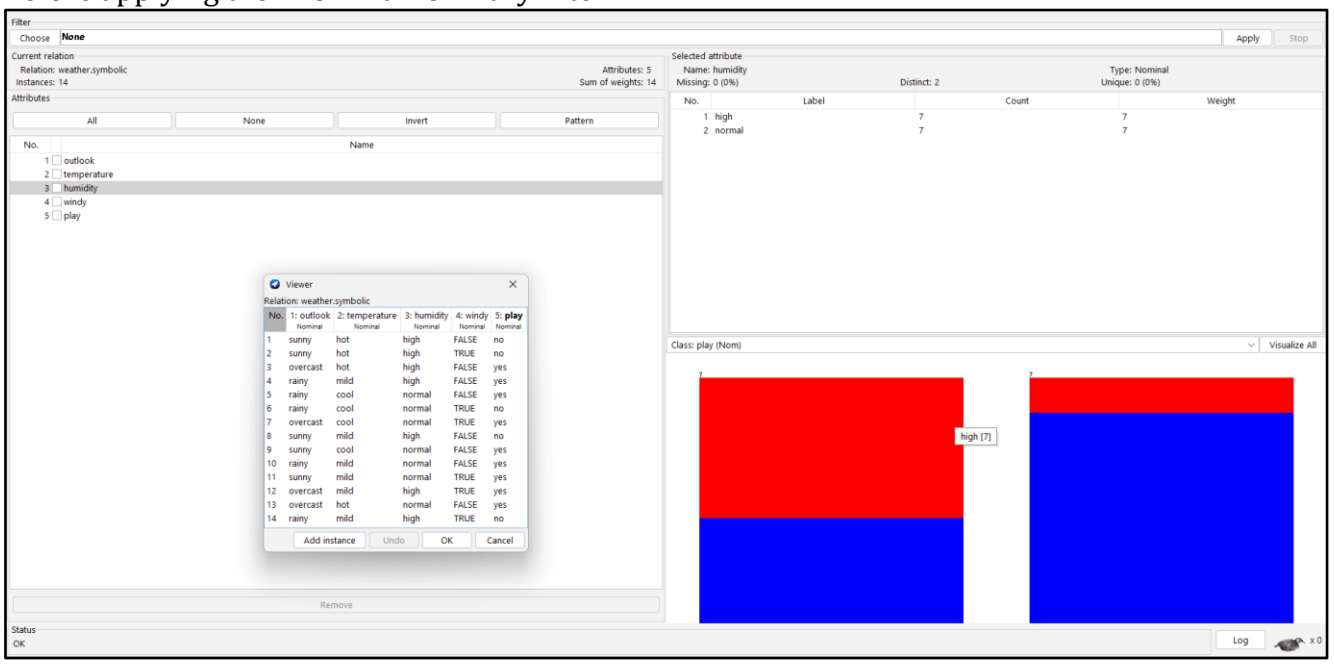

Fig 5.10

Description: Categorical labels used directly by algorithms like decision trees and rule-based systems. In this form, humidity is treated as a single attribute with two states. Classifiers use internal logic to distinguish between high and normal but can't do mathematical computations or easily integrate it into numeric models.

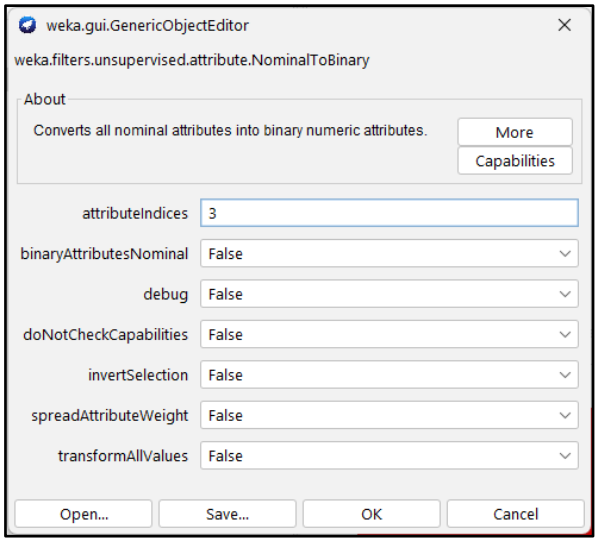

Fig 5.11

Filter Section click on choose button> Filter> Unsupervised> Select NominalToBinary filter.

After applying the "NominalToBinary filter"

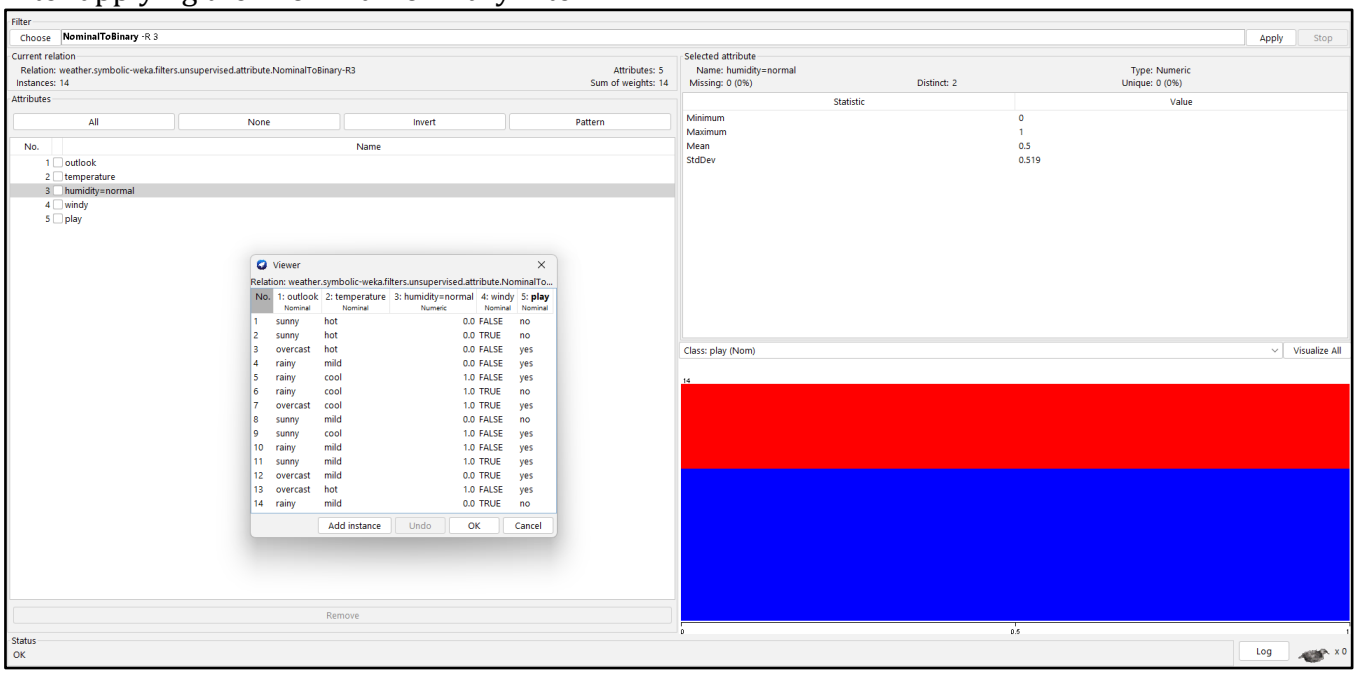

Fig 5.12

Description: The nominal attribute humidity is converted into binary format (one-hot encoded). A new binary attribute is created, such as humidity=high (values: 1 if cool/mild, 0 otherwise) The original humidity attribute is removed. Now, the dataset contains an explicit binary column instead of a multi-valued nominal.

4. Normalize filter: The Normalize filter is a unsupervised attribute filter used to scale numeric attributes in a dataset so that their values fall within a specified range, typically 0 to 1. This normalization process ensures that all numeric features contribute equally during model training, preventing attributes with larger values from dominating those with smaller ranges.

Choose Dataset: diabetes

Fig 5.13

Analysis: The diabetes dataset consists of nine attributes, where eight are numeric input features such as number of pregnancies (preg), glucose concentration (plas), blood pressure (pres), BMI (mass), and so on. The last attribute, class, is the target variable indicating whether a person is diabetic (1) or not (0). Since all input attributes are numerical, the dataset is well-suited for various machine learning algorithms like Logistic Regression, KNN, SVM, and Decision Trees. However, due to the different scales of attributes, preprocessing techniques such as normalization are important to ensure balanced model learning.

Before applying the “Normalize filter”

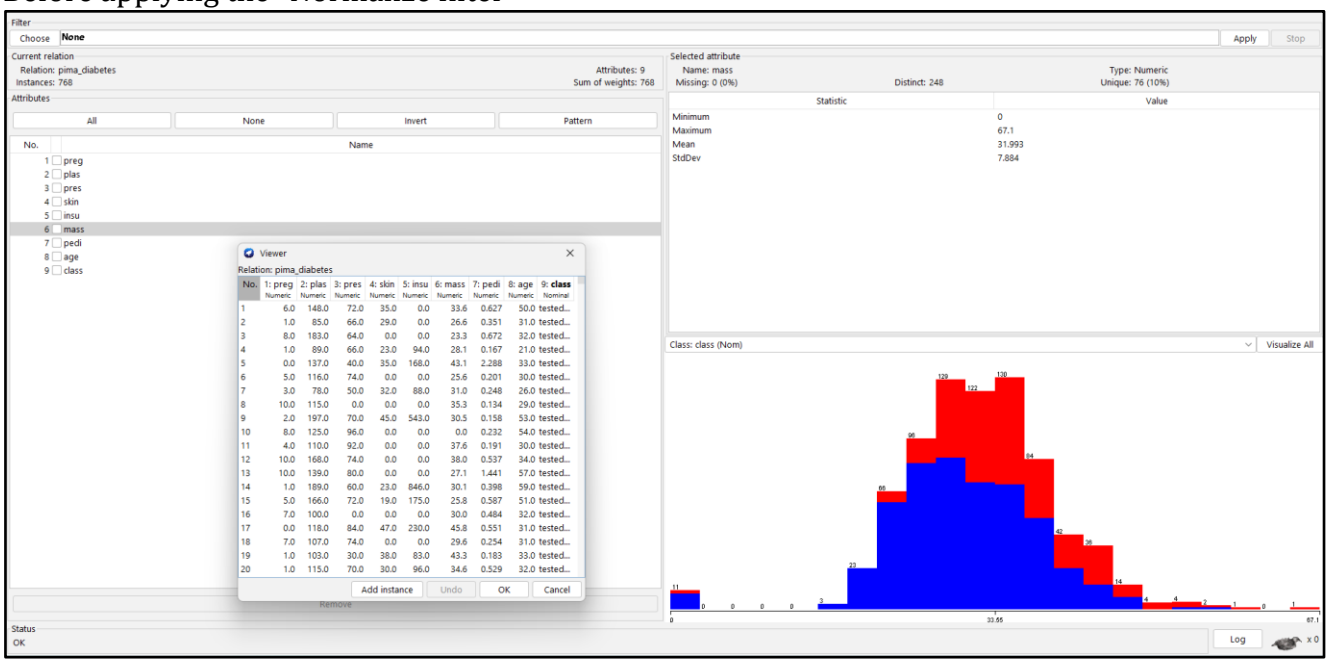

Fig 5.14

Description: Before normalization, all attributes in the diabetes dataset—such as preg, plas, mass, and age—have different value ranges. For example, age may range up to 80, while pedi may range only between 0 and 2. This inconsistency in scale can cause machine learning models, especially those based on distance or gradients, to be biased toward attributes with larger numeric ranges.

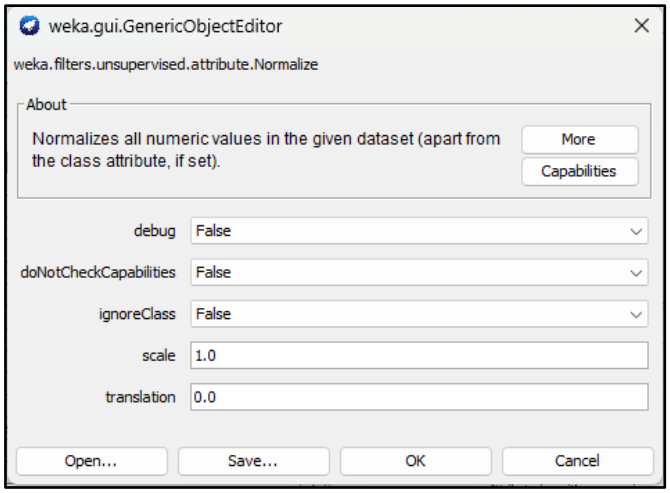

Fig 5.15

Filter Section click on choose button> Filter> Unsupervised> Select Normalize filter.

After applying the "Normalize filter"

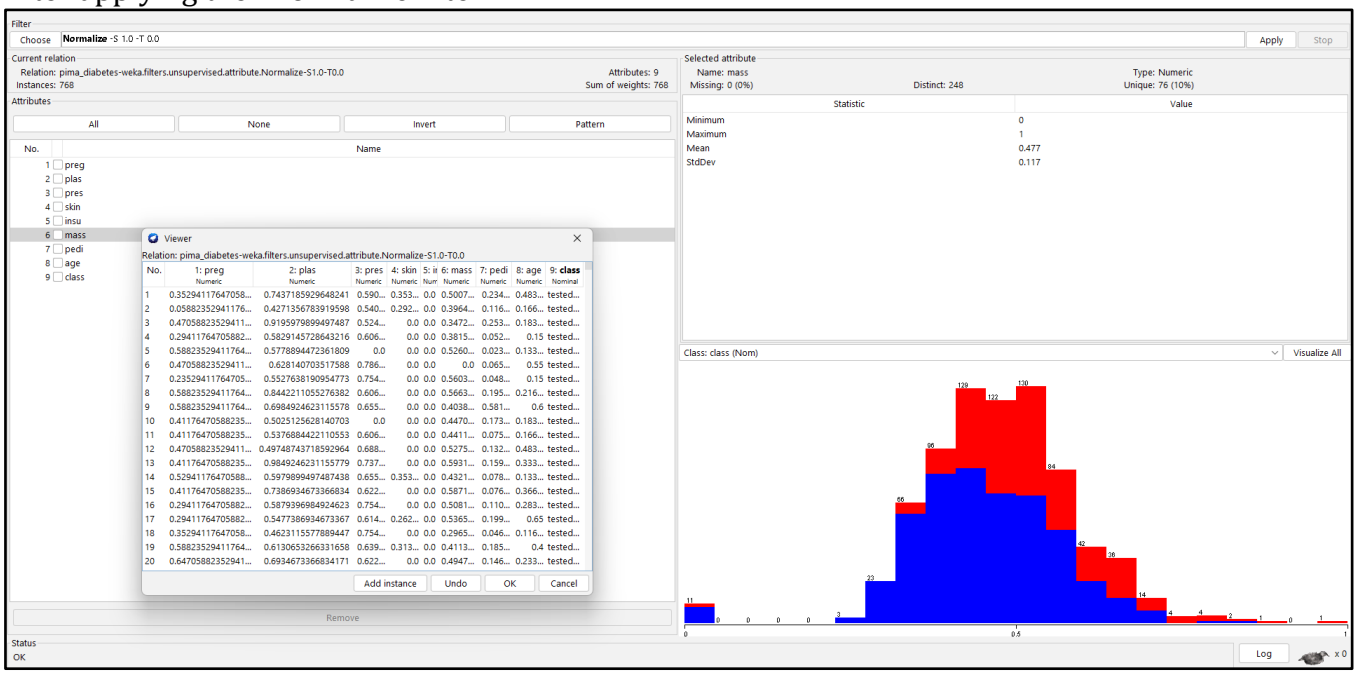

Fig 5.16

Description: After normalization, all numeric attributes are rescaled to a uniform range between 0 and 1. This ensures that each feature contributes equally during model training, improving performance for algorithms like KNN, SVM, and Neural Networks. Normalization makes the dataset more balanced, stable, and suitable for accurate classification.

Experiment Outcome:

The experiment focused on applying key preprocessing techniques using WEKA filters: NumericToNominal, StringToNominal, NominalToBinary, and Normalize to transform a raw dataset into a format suitable for machine learning models. These filters were applied sequentially to observe their effect on the structure and semantics of the dataset. After preprocessing, the dataset showed better compatibility with ML algorithms, reduced noise, and improved uniformity of data types.

The comparison between the dataset before and after preprocessing revealed improvements in consistency, interpretability, and numerical scale, which are critical for training efficient and accurate models. The transformation also addressed data type mismatches and prepared categorical features for algorithms that require numeric input.