THE STATE UNIVERSITY OF ZANZIBAR
SCHOOL OF SOCIAL AND NATURAL SCIENCE
DEPARTMENT OF SOCIAL SCIENCE TUNGUU CAMPUS.

COURSE TITLE:	DATA MINING AND BUSINESS INTELLIGENCE
COURSE CODE:	INF 3202
NATURE OF WORK:	GROUP ASSIGNMENT
STUDENTS NAME	REG NO.
ALI SIMAI ALI	BITAM/9/21/033/TZ
LECTURER:	DR. OMAR H. KOMBO

QUESTION: Use Orange Appache Mahout data mining tool with any dataset to predict the target variable using K-Nearest Neighbor (KNN) algorithm.

Orange Is a open source software used for data visualization , machine learning and data mining tool.
Orange give us a graphical user interface to orange's data mining and machine learning techniques.
Orange its support classification, regression, association rules and clustering a set of widgets for model for making data analysis for make prediction and filtering the data.
In orange can you different algorithm for making prediction of values or data among of them is **K-NEAREST NEIGHBOR (KNN) ALGORITHM.**
Which is used to Predict according to the nearest training instances which accept input of data set and provide the output according to the trained model use.
In orange provide **kNN** widget which use KNN algorithm to search data of K , set number of nearest neighbors,the distance parameter (metric) and weights as model criteria.
Distance parameter either (metric) can be distance between two points,straight line (Euclidean),sum of absolute differences of all attributes(Manhattan),greatest of absolute differences between attributes (Maximal) and distance between point and distribution(Mahalanobis).
And the *Weights* can be
Uniform: all points in each neighborhood are weighted equally.
Distance: closer neighbors of a query point have a greater influence than the neighbors further away.
KNN uses default preprocessing when no other preprocessors are given to run. It executes dataset in the following order:

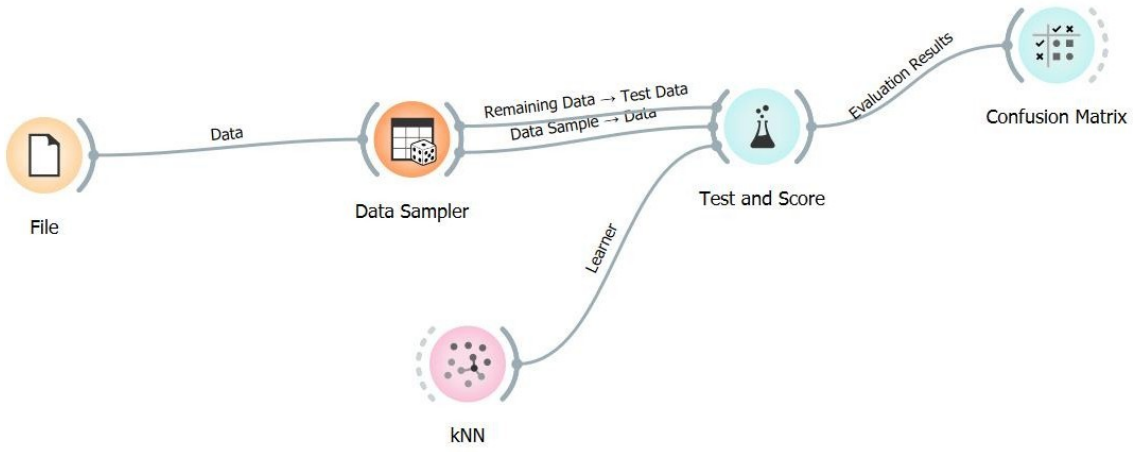
- removes instances with unknown target values
- continues categorical variables
- filter and removes empty columns
- imputes missing values with mean values
- Normalize the data

Example Workflow in Orange

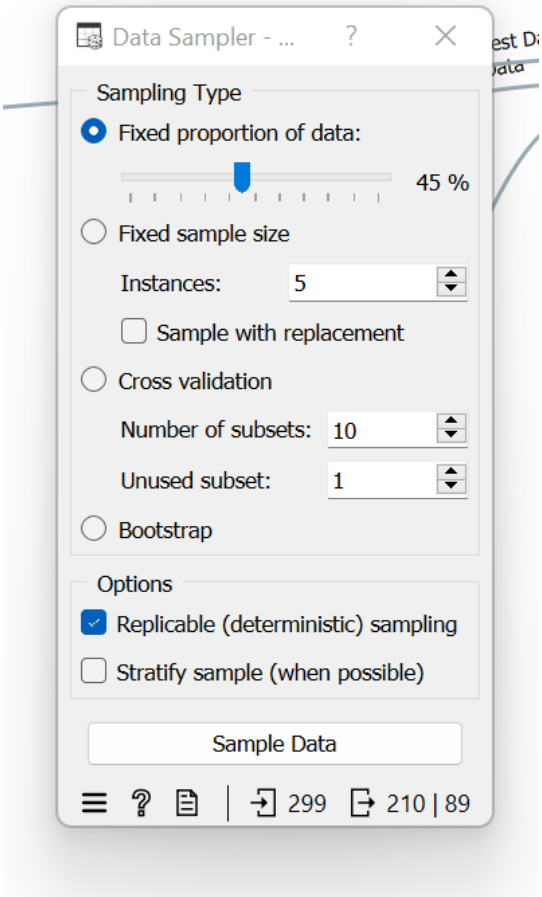
- ✓ File Widget: Load the dataset , by import or create new dataset (e.g., Iris dataset).
- ✓ Data Table Widget: View and inspect the dataset.
- ✓ Select Columns Widget: Choose relevant features and the target variable.
- ✓ Normalize Widget: Normalize the data if necessary.

- ✓ kNN Widget: Configure and set the number of neighbors (K) and the distance metric.
- ✓ Test & Score Widget: Connect the kNN widget to evaluate the model using cross- validation.
- ✓ Confusion Matrix Widget: Visualize the results.

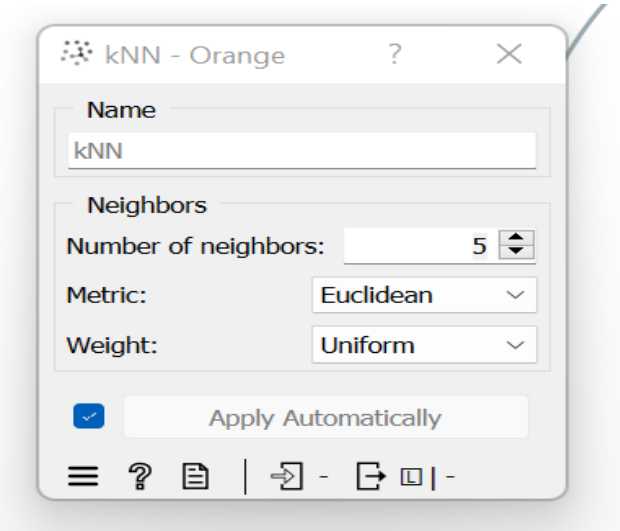
The following is the prediction according to the nearest training instances. Making prediction of death event using knn in orange.



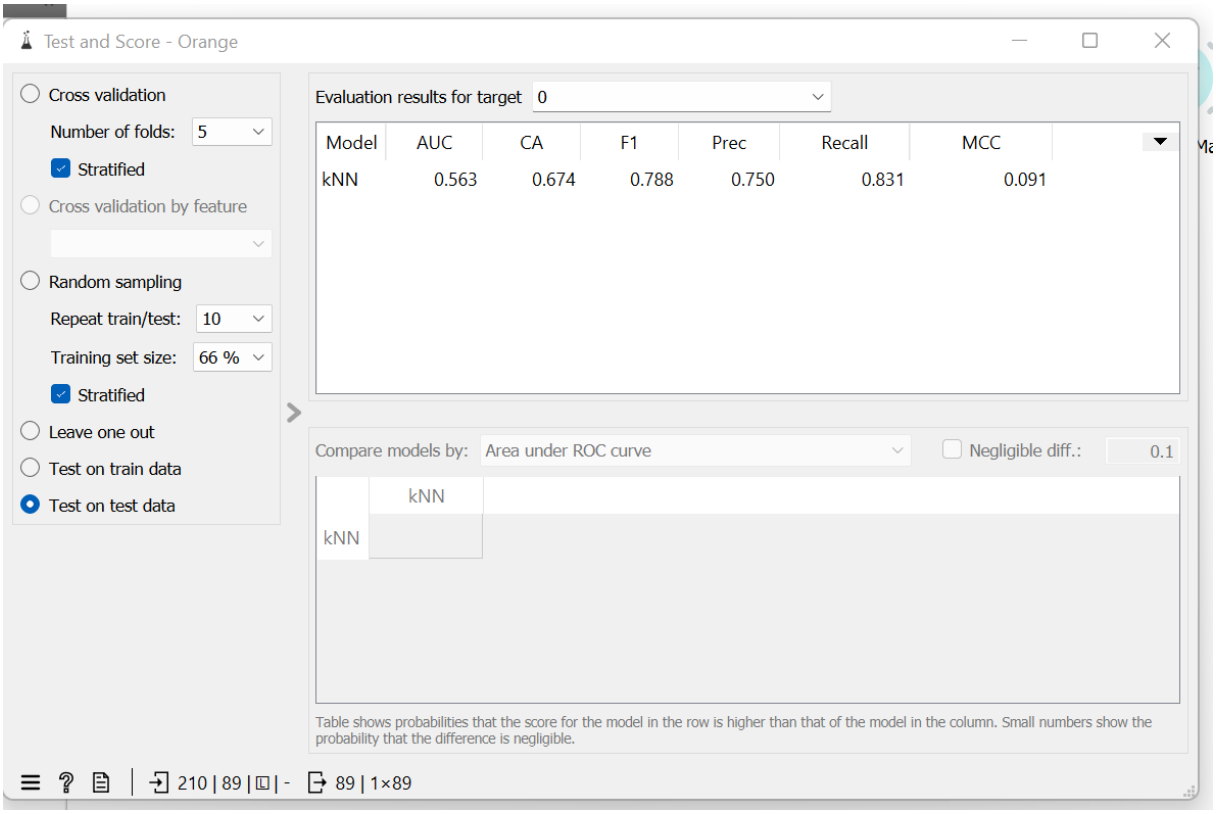
Data splitting.



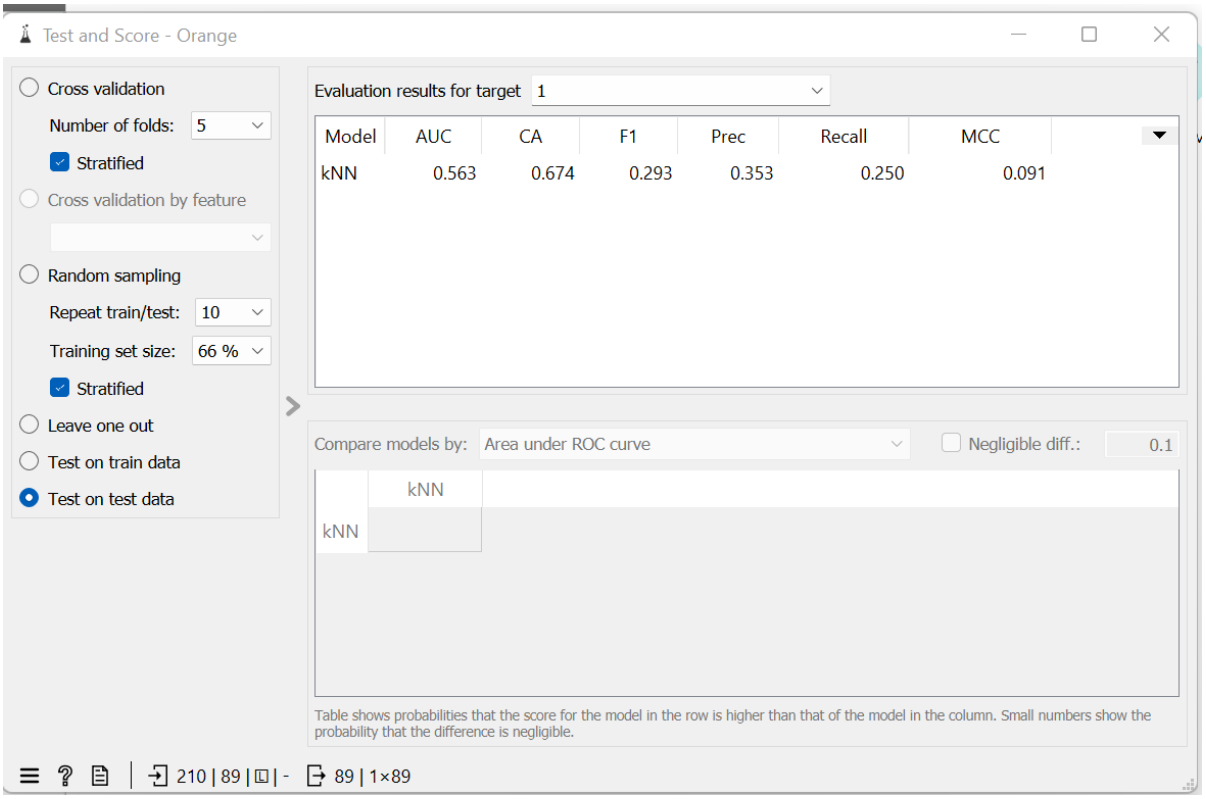
Set Knn Alorightm



Set Result for target 0



Result of target 1.



Part Two Assigment

`pd.to_numeric`: This is a function from the pandas library that attempts to convert values to numeric (integers or floats).

`errors='coerce'`: This parameter specifies how to handle errors during the conversion process.
`mean()`:

- This is a method in pandas that calculates the mean (average) of the numerical values in a Series (a single column in a Data Frame).

`.fillna()`: This is a pandas method used to fill missing values (`NaN`) in a Series (or Data Frame) with a specified value.

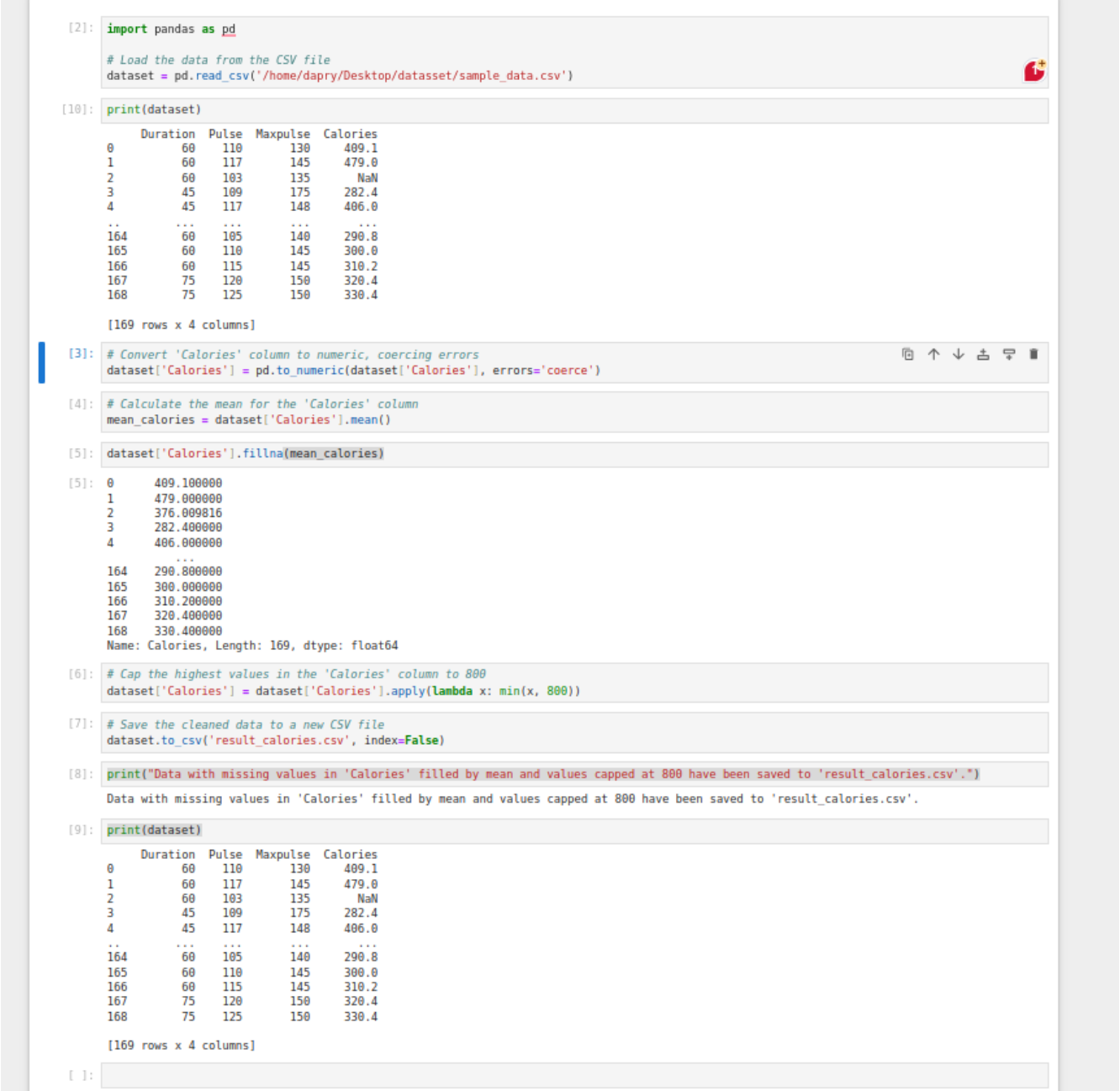
`apply(lambda x: min(x, 800))`:

- `.apply()`: This is a pandas method that applies a function to each element of a Series (or Data Frame). In this case, it applies a function to each value in the 'Calories' column.

- `lambda x: min(x, 800)`: This is an anonymous function (also known as a lambda function) that takes a single argument `x` and returns the minimum of `x` and 800.

`dataset.to_csv`:

- This is a method in pandas that is used to write the Data Frame to a CSV (Comma-Separated Values) file.



```
[2]: import pandas as pd

     # Load the data from the CSV file
     dataset = pd.read_csv('/home/dapry/Desktop/datasset/sample_data.csv')

[10]: print(dataset)

      Duration  Pulse  Maxpulse  Calories
0           60    110       130     409.1
1           60    117       145     479.0
2           60    103       135       NaN
3           45    109       175     282.4
4           45    117       148     406.0
..         ...    ...       ...       ...
164         60    105       140     290.8
165         60    110       145     300.0
166         60    115       145     310.2
167         75    120       150     320.4
168         75    125       150     330.4

[169 rows x 4 columns]

[3]: # Convert 'Calories' column to numeric, coercing errors
     dataset['Calories'] = pd.to_numeric(dataset['Calories'], errors='coerce')

[4]: # Calculate the mean for the 'Calories' column
     mean_calories = dataset['Calories'].mean()

[5]: dataset['Calories'].fillna(mean_calories)

[5]: 0      409.100000
     1      479.000000
     2      376.009816
     3      282.400000
     4      406.000000
               ...
     164    290.800000
     165    300.000000
     166    310.200000
     167    320.400000
     168    330.400000
     Name: Calories, Length: 169, dtype: float64

[6]: # Cap the highest values in the 'Calories' column to 800
     dataset['Calories'] = dataset['Calories'].apply(lambda x: min(x, 800))

[7]: # Save the cleaned data to a new CSV file
     dataset.to_csv('result_calories.csv', index=False)

[8]: print("Data with missing values in 'Calories' filled by mean and values capped at 800 have been saved to 'result_calories.csv'.")

     Data with missing values in 'Calories' filled by mean and values capped at 800 have been saved to 'result_calories.csv'.

[9]: print(dataset)

      Duration  Pulse  Maxpulse  Calories
0           60    110       130     409.1
1           60    117       145     479.0
2           60    103       135       NaN
3           45    109       175     282.4
4           45    117       148     406.0
..         ...    ...       ...       ...
164         60    105       140     290.8
165         60    110       145     300.0
166         60    115       145     310.2
167         75    120       150     320.4
168         75    125       150     330.4

[169 rows x 4 columns]

[ ]:
```