# Experiment 5

## Importing Libraries

import pandas as pd  
import numpy as np  
from sklearn.preprocessing import MinMaxScaler, StandardScaler  
import matplotlib.pyplot as plt  
import seaborn as sns

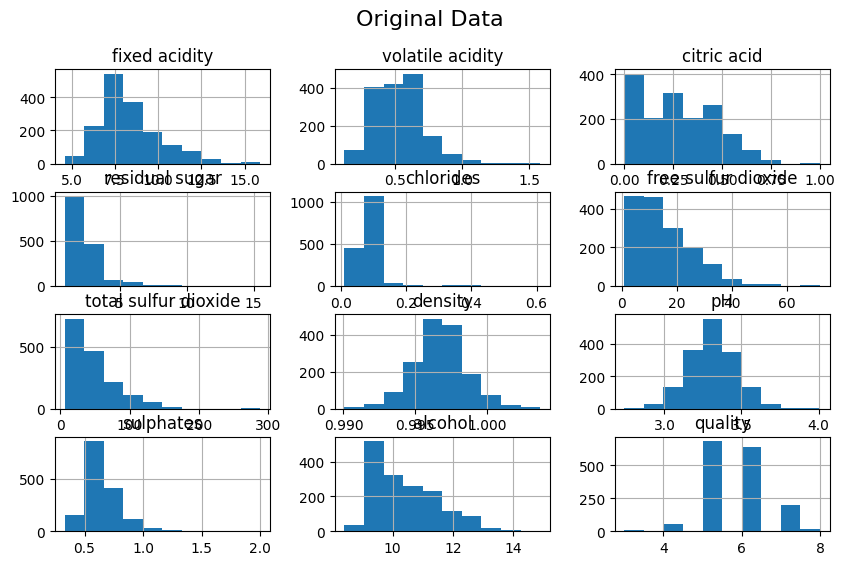
### Loading Dataset : [Wine Quality](https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv)

datasetUrl = 'https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv'  
df = pd.read\_csv(datasetUrl, sep=';')  
  
# Display the first five rows  
print(df.head())  
  
# Get information about data types and missing values  
print(df.info())  
  
# Summary statistics  
print(df.describe())

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 7.4 0.70 0.00 1.9 0.076   
1 7.8 0.88 0.00 2.6 0.098   
2 7.8 0.76 0.04 2.3 0.092   
3 11.2 0.28 0.56 1.9 0.075   
4 7.4 0.70 0.00 1.9 0.076   
  
 free sulfur dioxide total sulfur dioxide density pH sulphates \  
0 11.0 34.0 0.9978 3.51 0.56   
1 25.0 67.0 0.9968 3.20 0.68   
2 15.0 54.0 0.9970 3.26 0.65   
3 17.0 60.0 0.9980 3.16 0.58   
4 11.0 34.0 0.9978 3.51 0.56   
  
 alcohol quality   
0 9.4 5   
1 9.8 5   
2 9.8 5   
3 9.8 6   
4 9.4 5   
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1599 entries, 0 to 1598  
Data columns (total 12 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 fixed acidity 1599 non-null float64  
 1 volatile acidity 1599 non-null float64  
 2 citric acid 1599 non-null float64  
 3 residual sugar 1599 non-null float64  
 4 chlorides 1599 non-null float64  
 5 free sulfur dioxide 1599 non-null float64  
 6 total sulfur dioxide 1599 non-null float64  
 7 density 1599 non-null float64  
 8 pH 1599 non-null float64  
 9 sulphates 1599 non-null float64  
 10 alcohol 1599 non-null float64  
 11 quality 1599 non-null int64   
dtypes: float64(11), int64(1)  
memory usage: 150.0 KB  
None  
 fixed acidity volatile acidity citric acid residual sugar \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 8.319637 0.527821 0.270976 2.538806   
std 1.741096 0.179060 0.194801 1.409928   
min 4.600000 0.120000 0.000000 0.900000   
25% 7.100000 0.390000 0.090000 1.900000   
50% 7.900000 0.520000 0.260000 2.200000   
75% 9.200000 0.640000 0.420000 2.600000   
max 15.900000 1.580000 1.000000 15.500000   
  
 chlorides free sulfur dioxide total sulfur dioxide density \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.087467 15.874922 46.467792 0.996747   
std 0.047065 10.460157 32.895324 0.001887   
min 0.012000 1.000000 6.000000 0.990070   
25% 0.070000 7.000000 22.000000 0.995600   
50% 0.079000 14.000000 38.000000 0.996750   
75% 0.090000 21.000000 62.000000 0.997835   
max 0.611000 72.000000 289.000000 1.003690   
  
 pH sulphates alcohol quality   
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 3.311113 0.658149 10.422983 5.636023   
std 0.154386 0.169507 1.065668 0.807569   
min 2.740000 0.330000 8.400000 3.000000   
25% 3.210000 0.550000 9.500000 5.000000   
50% 3.310000 0.620000 10.200000 6.000000   
75% 3.400000 0.730000 11.100000 6.000000   
max 4.010000 2.000000 14.900000 8.000000

#### Plot Original Data

df.hist(bins=10, figsize=(10, 6))  
plt.suptitle("Original Data", fontsize=16)  
plt.show()



### Normalization Techniques

#### Min-Max Normalization

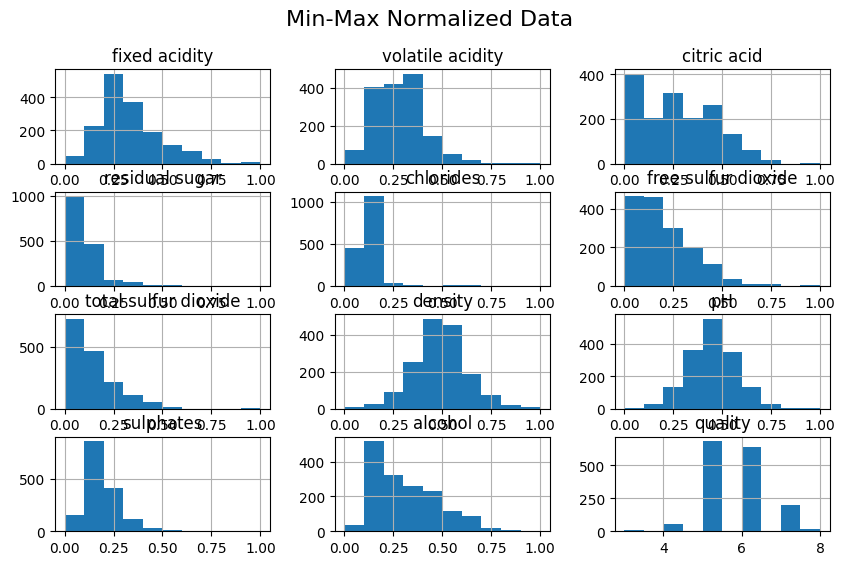
Scales the data to a fixed range, typically [0, 1].

# Initialize the MinMaxScaler  
min\_max\_scaler = MinMaxScaler()  
  
# Apply Min-Max normalization to all features except the target  
df\_min\_max = df.copy()  
df\_min\_max.iloc[:, :-1] = min\_max\_scaler.fit\_transform(df\_min\_max.iloc[:, :-1])  
  
df\_min\_max

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 0.247788 0.397260 0.00 0.068493 0.106845   
1 0.283186 0.520548 0.00 0.116438 0.143573   
2 0.283186 0.438356 0.04 0.095890 0.133556   
3 0.584071 0.109589 0.56 0.068493 0.105175   
4 0.247788 0.397260 0.00 0.068493 0.106845   
... ... ... ... ... ...   
1594 0.141593 0.328767 0.08 0.075342 0.130217   
1595 0.115044 0.294521 0.10 0.089041 0.083472   
1596 0.150442 0.267123 0.13 0.095890 0.106845   
1597 0.115044 0.359589 0.12 0.075342 0.105175   
1598 0.123894 0.130137 0.47 0.184932 0.091820   
  
 free sulfur dioxide total sulfur dioxide density pH \  
0 0.140845 0.098940 0.567548 0.606299   
1 0.338028 0.215548 0.494126 0.362205   
2 0.197183 0.169611 0.508811 0.409449   
3 0.225352 0.190813 0.582232 0.330709   
4 0.140845 0.098940 0.567548 0.606299   
... ... ... ... ...   
1594 0.436620 0.134276 0.354626 0.559055   
1595 0.535211 0.159011 0.370778 0.614173   
1596 0.394366 0.120141 0.416300 0.535433   
1597 0.436620 0.134276 0.396476 0.653543   
1598 0.239437 0.127208 0.397944 0.511811   
  
 sulphates alcohol quality   
0 0.137725 0.153846 5   
1 0.209581 0.215385 5   
2 0.191617 0.215385 5   
3 0.149701 0.215385 6   
4 0.137725 0.153846 5   
... ... ... ...   
1594 0.149701 0.323077 5   
1595 0.257485 0.430769 6   
1596 0.251497 0.400000 6   
1597 0.227545 0.276923 5   
1598 0.197605 0.400000 6   
  
[1599 rows x 12 columns]

#### Plot the Min Max Normalization

df\_min\_max.hist(bins=10, figsize=(10, 6))  
plt.suptitle("Min-Max Normalized Data", fontsize=16)  
plt.show()



#### Z-Score Normalization

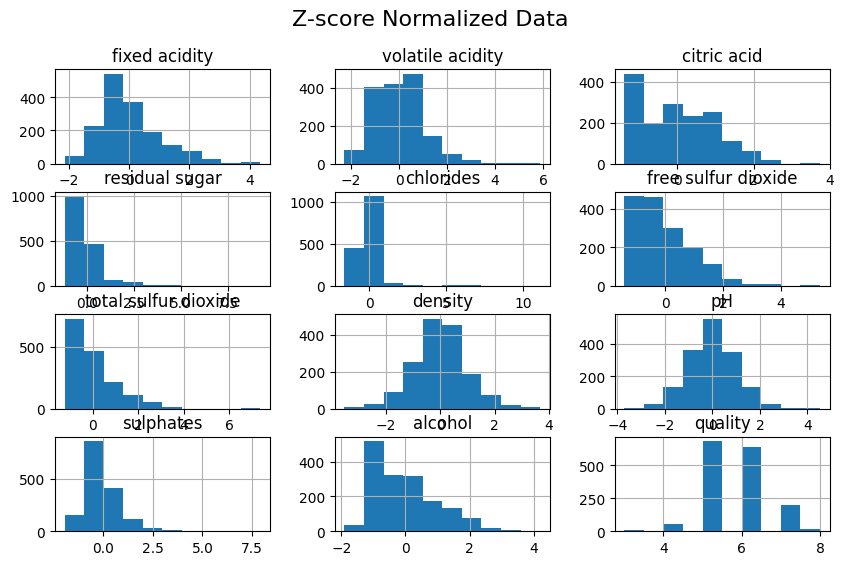
Transforms the data to have a mean of 0 and a standard deviation of 1.

# Initialize the StandardScaler  
z\_score\_scaler = StandardScaler()  
  
# Apply Z-score normalization to all features except the target  
df\_z\_score = df.copy()  
df\_z\_score.iloc[:, :-1] = z\_score\_scaler.fit\_transform(df\_z\_score.iloc[:, :-1])  
  
df\_z\_score

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 -0.528360 0.961877 -1.391472 -0.453218 -0.243707   
1 -0.298547 1.967442 -1.391472 0.043416 0.223875   
2 -0.298547 1.297065 -1.186070 -0.169427 0.096353   
3 1.654856 -1.384443 1.484154 -0.453218 -0.264960   
4 -0.528360 0.961877 -1.391472 -0.453218 -0.243707   
... ... ... ... ... ...   
1594 -1.217796 0.403229 -0.980669 -0.382271 0.053845   
1595 -1.390155 0.123905 -0.877968 -0.240375 -0.541259   
1596 -1.160343 -0.099554 -0.723916 -0.169427 -0.243707   
1597 -1.390155 0.654620 -0.775267 -0.382271 -0.264960   
1598 -1.332702 -1.216849 1.021999 0.752894 -0.434990   
  
 free sulfur dioxide total sulfur dioxide density pH \  
0 -0.466193 -0.379133 0.558274 1.288643   
1 0.872638 0.624363 0.028261 -0.719933   
2 -0.083669 0.229047 0.134264 -0.331177   
3 0.107592 0.411500 0.664277 -0.979104   
4 -0.466193 -0.379133 0.558274 1.288643   
... ... ... ... ...   
1594 1.542054 -0.075043 -0.978765 0.899886   
1595 2.211469 0.137820 -0.862162 1.353436   
1596 1.255161 -0.196679 -0.533554 0.705508   
1597 1.542054 -0.075043 -0.676657 1.677400   
1598 0.203223 -0.135861 -0.666057 0.511130   
  
 sulphates alcohol quality   
0 -0.579207 -0.960246 5   
1 0.128950 -0.584777 5   
2 -0.048089 -0.584777 5   
3 -0.461180 -0.584777 6   
4 -0.579207 -0.960246 5   
... ... ... ...   
1594 -0.461180 0.072294 5   
1595 0.601055 0.729364 6   
1596 0.542042 0.541630 6   
1597 0.305990 -0.209308 5   
1598 0.010924 0.541630 6   
  
[1599 rows x 12 columns]

#### Plot Z-Score Normalization Data

df\_z\_score.hist(bins=10, figsize=(10, 6))  
plt.suptitle("Z-score Normalized Data", fontsize=16)  
plt.show()



#### Decimal Scaling

Moves the decimal point of values to bring them within a certain range, typically [-1, 1].

# Function for Decimal Scaling  
def decimal\_scaling(df):  
 df\_decimal = df.copy()  
 for column in df\_decimal.columns[:-1]:  
 max\_abs = df\_decimal[column].abs().max()  
 j = np.ceil(np.log10(max\_abs + 1))  
 df\_decimal[column] = df\_decimal[column] / (10 \*\* j)  
 return df\_decimal  
  
# Apply Decimal Scaling  
df\_decimal = decimal\_scaling(df)  
  
df\_decimal

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 0.074 0.0700 0.000 0.019 0.0076   
1 0.078 0.0880 0.000 0.026 0.0098   
2 0.078 0.0760 0.004 0.023 0.0092   
3 0.112 0.0280 0.056 0.019 0.0075   
4 0.074 0.0700 0.000 0.019 0.0076   
... ... ... ... ... ...   
1594 0.062 0.0600 0.008 0.020 0.0090   
1595 0.059 0.0550 0.010 0.022 0.0062   
1596 0.063 0.0510 0.013 0.023 0.0076   
1597 0.059 0.0645 0.012 0.020 0.0075   
1598 0.060 0.0310 0.047 0.036 0.0067   
  
 free sulfur dioxide total sulfur dioxide density pH sulphates \  
0 0.11 0.034 0.099780 0.351 0.056   
1 0.25 0.067 0.099680 0.320 0.068   
2 0.15 0.054 0.099700 0.326 0.065   
3 0.17 0.060 0.099800 0.316 0.058   
4 0.11 0.034 0.099780 0.351 0.056   
... ... ... ... ... ...   
1594 0.32 0.044 0.099490 0.345 0.058   
1595 0.39 0.051 0.099512 0.352 0.076   
1596 0.29 0.040 0.099574 0.342 0.075   
1597 0.32 0.044 0.099547 0.357 0.071   
1598 0.18 0.042 0.099549 0.339 0.066   
  
 alcohol quality   
0 0.094 5   
1 0.098 5   
2 0.098 5   
3 0.098 6   
4 0.094 5   
... ... ...   
1594 0.105 5   
1595 0.112 6   
1596 0.110 6   
1597 0.102 5   
1598 0.110 6   
  
[1599 rows x 12 columns]

### Discretization Techniques

#### Binning (Equal Width)

Divides the range of the data into intervals of equal size.

# Define the number of bins  
num\_bins = 10  
  
# Apply Binning to all features except the target  
df\_binned = df.copy()  
for column in df\_binned.columns[:-1]:  
 df\_binned[column] = pd.cut(df\_binned[column], bins=num\_bins, labels=False)  
  
df\_binned

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 2 3 0 0 1   
1 2 5 0 1 1   
2 2 4 0 0 1   
3 5 1 5 0 1   
4 2 3 0 0 1   
... ... ... ... ... ...   
1594 1 3 0 0 1   
1595 1 2 0 0 0   
1596 1 2 1 0 1   
1597 1 3 1 0 1   
1598 1 1 4 1 0   
  
 free sulfur dioxide total sulfur dioxide density pH sulphates \  
0 1 0 5 6 1   
1 3 2 4 3 2   
2 1 1 5 4 1   
3 2 1 5 3 1   
4 1 0 5 6 1   
... ... ... ... .. ...   
1594 4 1 3 5 1   
1595 5 1 3 6 2   
1596 3 1 4 5 2   
1597 4 1 3 6 2   
1598 2 1 3 5 1   
  
 alcohol quality   
0 1 5   
1 2 5   
2 2 5   
3 2 6   
4 1 5   
... ... ...   
1594 3 5   
1595 4 6   
1596 3 6   
1597 2 5   
1598 3 6   
  
[1599 rows x 12 columns]

### Histogram-Based Binning

Uses the histogram of the data to determine bin edges, ensuring each bin has roughly the same number of samples.

# Equal Frequency Binning  
df\_hist\_binned = df.apply(lambda x: pd.qcut(x, q=5, labels=False, duplicates='drop'))  
  
df\_hist\_binned

fixed acidity volatile acidity citric acid residual sugar chlorides \  
0 1 4 0 1 1   
1 2 4 0 3 4   
2 2 4 0 2 3   
3 4 0 4 1 1   
4 1 4 0 1 1   
... ... ... ... ... ...   
1594 0 3 1 1 3   
1595 0 2 1 2 0   
1596 0 2 1 2 1   
1597 0 3 1 1 1   
1598 0 0 4 4 0   
  
 free sulfur dioxide total sulfur dioxide density pH sulphates \  
0 1 2 3 4 1   
1 4 3 2 1 3   
2 2 3 2 1 2   
3 3 3 3 0 1   
4 1 2 3 4 1   
... ... ... ... .. ...   
1594 4 2 0 4 1   
1595 4 3 0 4 3   
1596 4 2 1 3 3   
1597 4 2 1 4 3   
1598 3 2 1 3 3   
  
 alcohol quality   
0 0 0   
1 1 0   
2 1 0   
3 1 1   
4 0 0   
... ... ...   
1594 2 0   
1595 3 1   
1596 3 1   
1597 2 0   
1598 3 1   
  
[1599 rows x 12 columns]

### Analyzing the Effects of Different Techniques

We'll analyze the impact of normalization and discretization on:

1. **Type of Attributes**
2. **Statistical Parameters (Central Tendency and Dispersion)**
3. **Aptness of Proximity Metrics**

#### Type of Attributes

* **Normalization:** Does not change the type of attributes; they remain continuous.
* **Discretization:** Converts continuous attributes into categorical (ordinal) attributes.

#### Statistical Parameters

# Original Data Statistics  
print("Original Data Statistics:")  
print(df.describe())  
  
# Min-Max Normalized Data Statistics  
print("\nMin-Max Normalized Data Statistics:")  
print(df\_min\_max.describe())  
  
# Z-Score Normalized Data Statistics  
print("\nZ-Score Normalized Data Statistics:")  
print(df\_z\_score.describe())  
  
# Decimal Scaled Data Statistics  
print("\nDecimal Scaled Data Statistics:")  
print(df\_decimal.describe())

Original Data Statistics:  
 fixed acidity volatile acidity citric acid residual sugar \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 8.319637 0.527821 0.270976 2.538806   
std 1.741096 0.179060 0.194801 1.409928   
min 4.600000 0.120000 0.000000 0.900000   
25% 7.100000 0.390000 0.090000 1.900000   
50% 7.900000 0.520000 0.260000 2.200000   
75% 9.200000 0.640000 0.420000 2.600000   
max 15.900000 1.580000 1.000000 15.500000   
  
 chlorides free sulfur dioxide total sulfur dioxide density \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.087467 15.874922 46.467792 0.996747   
std 0.047065 10.460157 32.895324 0.001887   
min 0.012000 1.000000 6.000000 0.990070   
25% 0.070000 7.000000 22.000000 0.995600   
50% 0.079000 14.000000 38.000000 0.996750   
75% 0.090000 21.000000 62.000000 0.997835   
max 0.611000 72.000000 289.000000 1.003690   
  
 pH sulphates alcohol quality   
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 3.311113 0.658149 10.422983 5.636023   
std 0.154386 0.169507 1.065668 0.807569   
min 2.740000 0.330000 8.400000 3.000000   
25% 3.210000 0.550000 9.500000 5.000000   
50% 3.310000 0.620000 10.200000 6.000000   
75% 3.400000 0.730000 11.100000 6.000000   
max 4.010000 2.000000 14.900000 8.000000   
  
Min-Max Normalized Data Statistics:  
 fixed acidity volatile acidity citric acid residual sugar \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.329171 0.279329 0.270976 0.112247   
std 0.154079 0.122644 0.194801 0.096570   
min 0.000000 0.000000 0.000000 0.000000   
25% 0.221239 0.184932 0.090000 0.068493   
50% 0.292035 0.273973 0.260000 0.089041   
75% 0.407080 0.356164 0.420000 0.116438   
max 1.000000 1.000000 1.000000 1.000000   
  
 chlorides free sulfur dioxide total sulfur dioxide density \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.125988 0.209506 0.142996 0.490211   
std 0.078573 0.147326 0.116238 0.138571   
min 0.000000 0.000000 0.000000 0.000000   
25% 0.096828 0.084507 0.056537 0.406021   
50% 0.111853 0.183099 0.113074 0.490455   
75% 0.130217 0.281690 0.197880 0.570117   
max 1.000000 1.000000 1.000000 1.000000   
  
 pH sulphates alcohol quality   
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.449695 0.196496 0.311228 5.636023   
std 0.121564 0.101501 0.163949 0.807569   
min 0.000000 0.000000 0.000000 3.000000   
25% 0.370079 0.131737 0.169231 5.000000   
50% 0.448819 0.173653 0.276923 6.000000   
75% 0.519685 0.239521 0.415385 6.000000   
max 1.000000 1.000000 1.000000 8.000000   
  
Z-Score Normalized Data Statistics:  
 fixed acidity volatile acidity citric acid residual sugar \  
count 1.599000e+03 1.599000e+03 1.599000e+03 1.599000e+03   
mean 3.554936e-16 1.733031e-16 -8.887339e-17 -1.244227e-16   
std 1.000313e+00 1.000313e+00 1.000313e+00 1.000313e+00   
min -2.137045e+00 -2.278280e+00 -1.391472e+00 -1.162696e+00   
25% -7.007187e-01 -7.699311e-01 -9.293181e-01 -4.532184e-01   
50% -2.410944e-01 -4.368911e-02 -5.636026e-02 -2.403750e-01   
75% 5.057952e-01 6.266881e-01 7.652471e-01 4.341614e-02   
max 4.355149e+00 5.877976e+00 3.743574e+00 9.195681e+00   
  
 chlorides free sulfur dioxide total sulfur dioxide density \  
count 1.599000e+03 1.599000e+03 1.599000e+03 1.599000e+03   
mean 3.732682e-16 -6.221137e-17 4.443669e-17 -3.473172e-14   
std 1.000313e+00 1.000313e+00 1.000313e+00 1.000313e+00   
min -1.603945e+00 -1.422500e+00 -1.230584e+00 -3.538731e+00   
25% -3.712290e-01 -8.487156e-01 -7.440403e-01 -6.077557e-01   
50% -1.799455e-01 -1.793002e-01 -2.574968e-01 1.760083e-03   
75% 5.384542e-02 4.901152e-01 4.723184e-01 5.768249e-01   
max 1.112703e+01 5.367284e+00 7.375154e+00 3.680055e+00   
  
 pH sulphates alcohol quality   
count 1.599000e+03 1.599000e+03 1.599000e+03 1599.000000   
mean 2.861723e-15 6.754377e-16 1.066481e-16 5.636023   
std 1.000313e+00 1.000313e+00 1.000313e+00 0.807569   
min -3.700401e+00 -1.936507e+00 -1.898919e+00 3.000000   
25% -6.551405e-01 -6.382196e-01 -8.663789e-01 5.000000   
50% -7.212705e-03 -2.251281e-01 -2.093081e-01 6.000000   
75% 5.759223e-01 4.240158e-01 6.354971e-01 6.000000   
max 4.528282e+00 7.918677e+00 4.202453e+00 8.000000   
  
Decimal Scaled Data Statistics:  
 fixed acidity volatile acidity citric acid residual sugar \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.083196 0.052782 0.027098 0.025388   
std 0.017411 0.017906 0.019480 0.014099   
min 0.046000 0.012000 0.000000 0.009000   
25% 0.071000 0.039000 0.009000 0.019000   
50% 0.079000 0.052000 0.026000 0.022000   
75% 0.092000 0.064000 0.042000 0.026000   
max 0.159000 0.158000 0.100000 0.155000   
  
 chlorides free sulfur dioxide total sulfur dioxide density \  
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.008747 0.158749 0.046468 0.099675   
std 0.004707 0.104602 0.032895 0.000189   
min 0.001200 0.010000 0.006000 0.099007   
25% 0.007000 0.070000 0.022000 0.099560   
50% 0.007900 0.140000 0.038000 0.099675   
75% 0.009000 0.210000 0.062000 0.099783   
max 0.061100 0.720000 0.289000 0.100369   
  
 pH sulphates alcohol quality   
count 1599.000000 1599.000000 1599.000000 1599.000000   
mean 0.331111 0.065815 0.104230 5.636023   
std 0.015439 0.016951 0.010657 0.807569   
min 0.274000 0.033000 0.084000 3.000000   
25% 0.321000 0.055000 0.095000 5.000000   
50% 0.331000 0.062000 0.102000 6.000000   
75% 0.340000 0.073000 0.111000 6.000000   
max 0.401000 0.200000 0.149000 8.000000

### **Aptness of Proximity Metrics**

* **Without Normalization:**
  + Features with larger ranges dominate distance calculations, potentially biasing models.
* **With Normalization:**
  + Ensures each feature contributes equally, leading to more balanced and meaningful distance metrics.

**Implications:**

* **K-Nearest Neighbors (KNN):** Performance significantly improves with normalization.
* **Clustering Algorithms (e.g., K-Means):** Better cluster formation due to balanced feature contributions.

### **Discretization Effects**

* **Binning (Equal Width):**
  + Simple to implement but may not account for data distribution.
* **Histogram-Based Binning:**
  + Accounts for data distribution, providing more balanced bins.

**Implications:**

* **Decision Trees:** Discretization can lead to simpler tree structures.
* **Reduced Information:** May lose some information due to grouping, potentially impacting model performance.

## **Conclusion**

Normalization and discretization are crucial preprocessing steps in machine learning workflows. They address issues related to varying feature scales and continuous data representation, respectively. Here's a summary of the key takeaways from the experiment:

1. **Normalization:**
   * Essential for algorithms sensitive to feature scales.
   * Techniques like Min-Max and Z-score normalization adjust data scales effectively.
   * Z-score normalization is particularly useful for standardizing data distributions.
2. **Discretization:**
   * Transforms continuous data into categorical bins, suitable for certain algorithms.
   * Binning methods can simplify data but may lead to information loss.
   * Histogram-based binning provides a more balanced approach by considering data distribution.
3. **Impact on Proximity Metrics:**
   * Normalization ensures equitable feature contribution in distance calculations.
   * Enhances the performance of distance-based algorithms like KNN and clustering.
4. **Statistical Parameters:**
   * Normalization alters central tendency and dispersion, making data suitable for various algorithms.
   * Discretization affects the representation of data, converting continuous attributes into categorical ones.

**Final Recommendation:**

* **Before Applying Normalization:**
  + Assess the algorithms to be used and their sensitivity to feature scales.
  + Choose normalization techniques that align with the model requirements.
* **Before Applying Discretization:**
  + Determine if the model benefits from categorical representation.
  + Select appropriate binning methods based on data distribution and model needs.

In summary, thoughtful application of normalization and discretization enhances model performance, ensures balanced feature contributions, and aligns data representation with algorithmic requirements.