Department of Electronic & Telecommunication Engineering
University of Moratuwa

EN3150 - PATTERN RECOGNITION

HOMEWORK 01 DATA PRE-PROCESSING

MANIMOHAN T. *200377M*

This report is submitted as Assignment of module EN3150
$21^{st} August 2023$

1 DATA PRE-PROCESSING

Task: Comparing of different data normalization methods.
Github link :- Codes and Outputs

1.1 Visit the following web page to get an idea about the available data sets in scikit learn and how to load them.

https://scikit-learn.org/stable/datasets.html

Selected dataset:- Wine recognition dataset

1.2 Load an available dataset of your choice. For example, you can load the Iris dataset (Since, we have used California housing dataset in the class, please do not use it)

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import load_wine
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import to_categorical

# Load the wine dataset
wine = load_wine()
X, y = wine.data, wine.target

#Print the class names
class_names = wine.target_names
print("Class Names:")
for i, name in enumerate(class_names):
    print(f"Class {i}: {name}")

# Get the feature names
feature_names = wine.feature_names
# Print the feature names
print("Feature Names:")
print(feature_names)
# Get the description of the dataset
description = wine.DESCR
```

```
# Print the description
print(description)
# Load the California housing dataset
```

1.3 Explore the dataset: Print the feature names, target variable (if applicable), and any relevant information about the dataset.

```
Class Names:
Class 0: class_0
Class 1: class_1
Class 2: class_2
Feature Names:
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols',
'flavanoids', 'nonflavanoid_phenols', 'proanthocyanins', 'color_intensity', 'hue',
'od280/od315_of_diluted_wines', 'proline']
.. _wine_dataset:
Wine recognition dataset
------------------------
**Data Set Characteristics:**
    :Number of Instances: 178
    :Number of Attributes: 13 numeric, predictive attributes and the class
    :Attribute Information:
  - Alcohol
  - Malic acid
  - Ash
 - Alcalinity of ash
  - Magnesium
 - Total phenols
  - Flavanoids
  - Nonflavanoid phenols
  - Proanthocyanins
...
  Tech. Rep. no. 92-01, (1992), Dept. of Computer Science and Dept. of
  Mathematics and Statistics, James Cook University of North Queensland.
  (Also submitted to Journal of Chemometrics)
```

Visualization

```
# Convert the dataset to a pandas DataFrame for easy visualization
df = pd.DataFrame(X, columns=wine.feature_names)
df['Class'] = wine.target_names[y]
# Plot pair plots with class colors
sns.pairplot(df, hue='Class', markers=['o', 's', '^'])
```

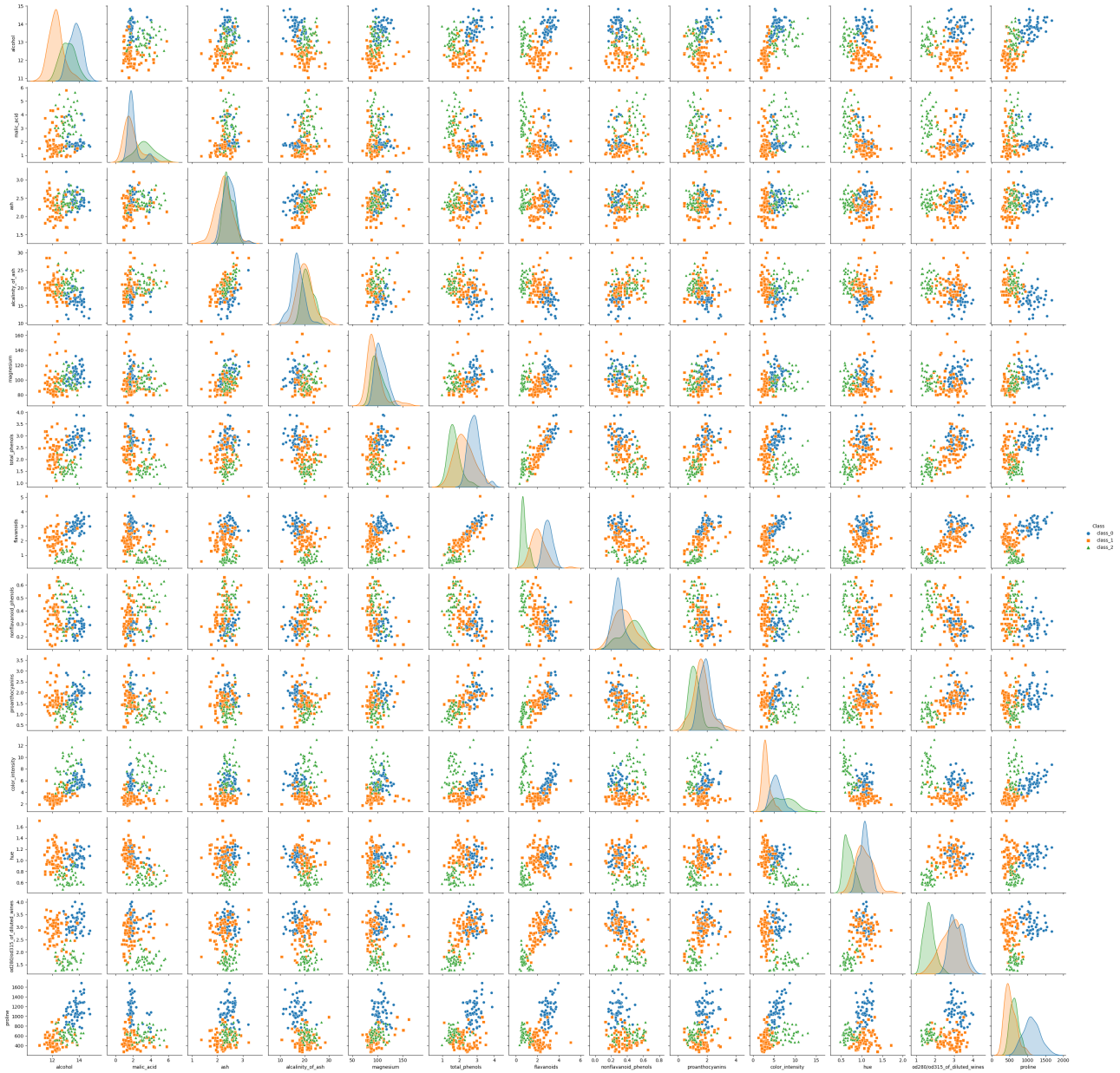

Figure 1 — Explore Data

```
plt.show()
```

1.4 Select the features: Choose a subset of features (e.g., say two features) from the dataset for the normalization comparison. [Hint: See the mean, variance, 1st quartile (25th quantile) and the 3rd quartile (75th quantile) of the features and select features that may contain outliers.]

```
# Choose a subset of features (e.g., first two features)
selected_features = X[:, :2]
print(selected_features)
```

```
[[14.23  1.71]
 [13.2   1.78]
 [13.16  2.36]
 [14.37  1.95]
```

```
 [13.24  2.59]
 [14.2   1.76]
 [14.39  1.87]
 [14.06  2.15]
 [14.83  1.64]
 [13.86  1.35]
 [14.1   2.16]
 [14.12  1.48]
 [13.75  1.73]
 [14.75  1.73]
 [14.38  1.87]
 [13.63  1.81]
 [14.3   1.92]
 [13.83  1.57]
 [14.19  1.59]
 [13.64  3.1 ]
 [14.06  1.63]
 [12.93  3.8 ]
 [13.71  1.86]
 [12.85  1.6 ]
 [13.5   1.81]
...
 [13.4   3.91]
 [13.27  4.28]
 [13.17  2.59]
 [14.13  4.1 ]]
```

1.5 Apply different normalization methods. Use the following normalization methods from 'sklearn.preprocessing':

- Min-Max Scaling (MinMaxScaler)
- Standard Scaling (StandardScaler)
- Robust Scaling (RobustScaler)
- Power Transformer

```
from sklearn.preprocessing import MinMaxScaler,StandardScaler,RobustScaler,PowerTransfor
# Apply normalization methods
methods = [MinMaxScaler(), StandardScaler(), RobustScaler(), PowerTransformer()]

normalized_data = [method.fit_transform(selected_features) for method in methods]
print(normalized_data )
```

1.6 Normalize the data:Apply each normalization method to the selected features.

```
 [array([[0.84210526, 0.1916996 ],
[0.57105263, 0.2055336 ],
[0.56052632, 0.3201581 ],
[0.87894737, 0.23913043],
[0.58157895, 0.36561265],
[0.83421053, 0.20158103],
[0.88421053, 0.22332016],
[0.79736842, 0.27865613],
[1.        , 0.17786561],
[0.74473684, 0.12055336],
[0.80789474, 0.28063241],
[0.81315789, 0.14624506],
[0.71578947, 0.19565217],
[0.97894737, 0.19565217],
[0.88157895, 0.22332016],
[0.68421053, 0.21146245],
[0.86052632, 0.23320158],
[0.73684211, 0.16403162],
[0.83157895, 0.16798419],
[0.68684211, 0.46640316],
[0.79736842, 0.17588933],
[0.5       , 0.60474308],
[0.70526316, 0.22134387],
[0.47894737, 0.16996047],
[0.65      , 0.21146245],
...
[ 0.48714794,  1.35342441],
[ 0.32533191,  1.51632354],
[ 0.2011548 ,  0.52369879],
[ 1.40378286,  1.43980882]])]
```

1.7 Visualize the data before and after normalization.Create scatter plots or histograms of the original and normalized data to visualize the effects of each normalization method on the feature distributions.

```
    # Visualize data before and after normalization
plt.figure(figsize=(12, 8))

plt.subplot(2, 3, 1)
plt.scatter(selected_features[:, 0], selected_features[:, 1])
plt.title("Original Data")

for i, method in enumerate(methods):
    plt.subplot(2, 3, i + 2)
    plt.scatter(normalized_data[i][:, 0], normalized_data[i][:, 1])
    plt.title(str(type(method))[17:-2])

plt.tight_layout()
plt.show()
```

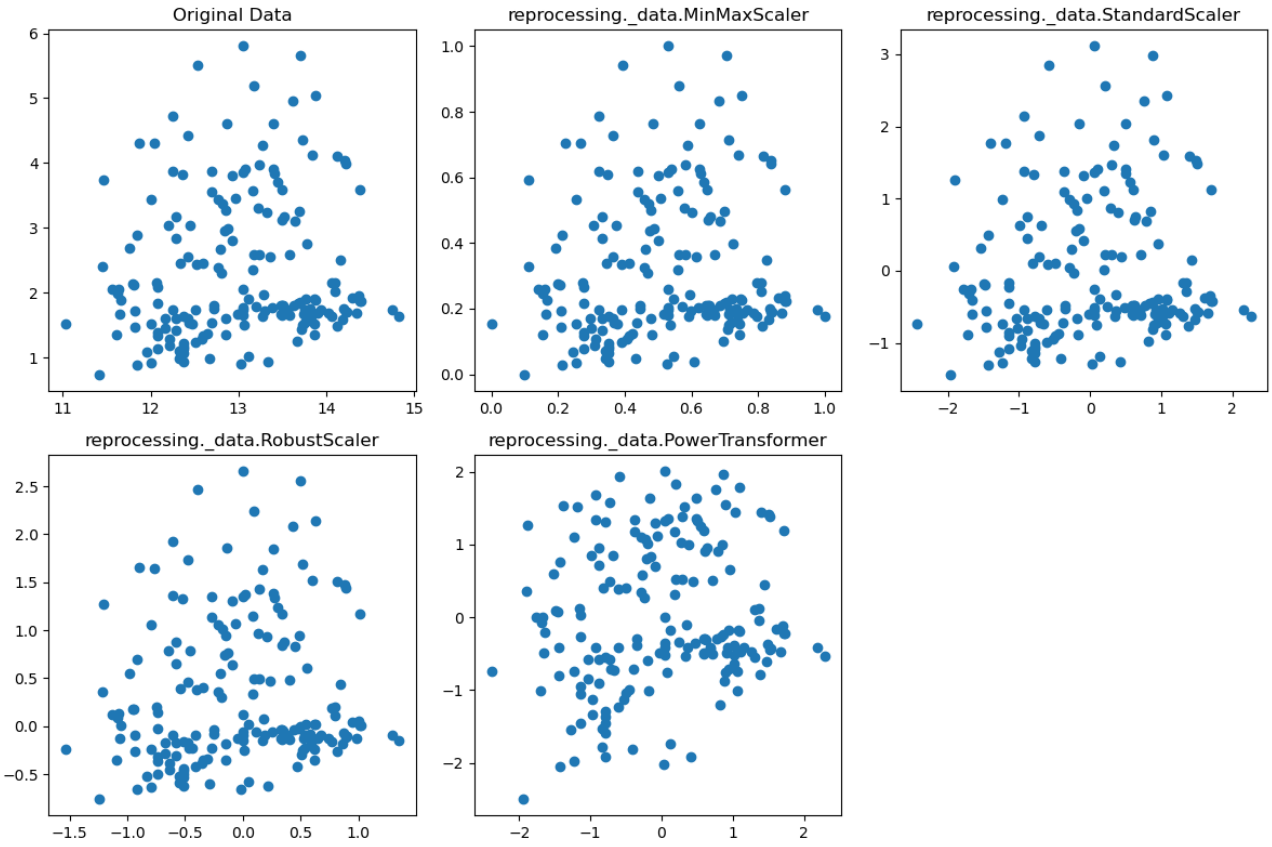

Figure 2 — Visualize the data before and after normalization

1.8 Compare how each normalization method scales the data and its impact on outliers.

1. Min-Max Scaling (MinMaxScaler):
   Min-Max Scaling transforms the data to a specific range (usually between 0 and 1) by linearly scaling each feature. It is given by the formula:

$$X_{Scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

   Impact on Outliers:
   Min-Max Scaling can be heavily affected by outliers, especially when the range of the data is dominated by the outliers.
   Outliers can significantly influence the scaling, causing most of the inlier data to be compressed into a small fraction of the scaled range.

2. Standard Scaling (StandardScaler):
   Standard Scaling transforms the data to have a mean of 0 and a standard deviation of 1. It is given by the formula:

$$X_{Scaled} = \frac{X - mean(X)}{std(X)} \tag{2}$$

   Impact on Outliers:
   Standard Scaling is less sensitive to outliers compared to Min-Max Scaling because it uses the mean and standard deviation, which are not heavily affected by outliers.
   Outliers can still have some influence, but they will have less impact on the scaling.

3. Robust Scaling (RobustScaler):
   Robust Scaling scales the data based on the median and the interquartile range (IQR), which makes it robust to outliers. It is given by the formula:

$$X_{Scaled} = \frac{X - median(X)}{IQR(X)} \tag{3}$$

   Impact on Outliers:
   Robust Scaling is designed to handle outliers well, as it uses the median and IQR, which are not influenced by extreme values.
   Outliers have a limited impact on the scaling, and the bulk of the data is transformed more consistently.

4. Power Transformer:
   Power Transformer is a family of methods that applies a power transformation to make the data more Gaussian-like. This can be useful when the data is not normally distributed.
   Impact on Outliers:
   The impact of outliers on power transformation depends on the specific transformation used (e.g., Box-Cox or Yeo-Johnson).
   Some power transformations can mitigate the impact of outliers by making the data more

Gaussian-like, while others may be influenced by extreme values.
In summary, the choice of normalization method should depend on the characteristics of your data, including the presence of outliers and the distribution of the data. Min-Max Scaling is sensitive to outliers, while Standard Scaling is less sensitive. Robust Scaling is specifically designed to handle outliers, and Power Transformation methods can be used to make the data more Gaussian-like, potentially reducing the impact of outliers on certain transformations. Always evaluate the performance of different normalization methods on your specific dataset to choose the one that suits your needs best.

1.9 Interpret the Results. Discuss the effects of each normalization method on the data's distribution, scale, and outlier handling. Which methods are more robust for outliers? Why?

The effects of each normalization method on the data's distribution, scale, and outlier handling are as follows:

* Min-Max Scaling (MinMaxScaler):
  -Scales the features to a specified range (usually [0, 1]).
  -Distributes the data uniformly within the specified range.
  -Does not handle outliers well as they can affect the scaling significantly.

* Standard Scaling (StandardScaler):
  -Scales the features to have a mean of 0 and a standard deviation of 1.
  -Assumes the data is normally distributed.
  -Handles outliers better than Min-Max scaling but can still be affected by extreme outliers.

* Robust Scaling (RobustScaler):
  -Scales the features using median and interquartile range, making it robust to outliers.
  -Performs well in the presence of outliers by scaling based on the data's central tendency.
  -Suitable when the data contains outliers.

* Power Transformer:
  -Applies power transformations to make the data more Gaussian-like.
  -Helps stabilize variance and make the data more suitable for algorithms that assume Gaussian distribution.
  -Tends to handle outliers better than Min-Max scaling and Standard scaling.

Which Methods are More Robust for Outliers? Why?

```
The methods that are more robust to outliers are RobustScaler and PowerTransformer.
*RobustScaler uses the median and interquartile range, making it less sensitive to
outliers.
*PowerTransformer transforms data to have a more Gaussian-like distribution, which
is less affected by outliers.
*Min-Max scaling and Standard scaling are more affected by outliers, as they are
```

influenced by the range and mean/std deviation, respectively.
*The choice of normalization method depends on the characteristics of the data and the requirements of your specific machine learning algorithm. If your data contains outliers and non-normal distributions, using methods like RobustScaler or PowerTransformer can help improve the robustness and performance of your models.

***