**A picture containing shape, arrow

Description automatically generatedDISCRETIZATION**

Instructions:

Please share your answers filled inline in the word document. Submit Python code and R code files wherever applicable.

Please ensure you update all the details:

**Name: Vishvash C**

**Batch Id: 23012024**

**Topic: Data Pre-Processing**

**Problem Statement:**

Everything will revolve around the data in Analytics world. Proper data will help you to make useful predictions that improve your business. Sometimes the usage of original data as it is does not help to have accurate solutions. It is needed to convert the data from one form to another form to have better predictions. Explore various techniques to transform the data for better model performance. you can go through this link:

<https://360digitmg.com/mindmap-data-science>

1. Convert the continuous data into discrete classes on the iris dataset.

Prepare the dataset by performing the preprocessing techniques, to have the data which improves model performance.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5 | 3.6 | 1.4 | 0.2 | setosa |
| 5.4 | 3.9 | 1.7 | 0.4 | setosa |
| 4.6 | 3.4 | 1.4 | 0.3 | setosa |
| 5 | 3.4 | 1.5 | 0.2 | setosa |
| 4.4 | 2.9 | 1.4 | 0.2 | setosa |
| 4.9 | 3.1 | 1.5 | 0.1 | setosa |

**Hints:**

For each assignment, the solution should be submitted in the below format.

1. **A picture containing shape, arrow

   Description automatically generated**Work on each feature to create a data dictionary as displayed in the image displayed below:
2. Hint: Refer to Iris.csv, which is a public dataset.
3. Research and perform all possible steps for obtaining the solution.
4. All the codes (executable programs) should execute without errors.
5. Code modularization should be followed.
6. Each line of code should have comments explaining the logic and why you are using that function.

import pandas as pd

df = pd.read\_csv(r"C:/Users/Lenovo/Downloads/Study material/EDA/InClass\_DataPreprocessing\_datasets/iris.csv")

df.info()

df\_num = df.iloc[:,1:5]

df\_num.corr()

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df\_fixed\_binning = df.copy()

df\_adaptive\_binning = df.copy()

# Fixed width binning

fixed\_bins = [0, 4.5, 6.5, 8.5] # Define bins manually

df\_fixed\_binning['Sepal.Length'] = pd.DataFrame(pd.cut(df['Sepal.Length'], bins=fixed\_bins, labels=['Small', 'Medium', 'Large']))

df\_fixed\_binning['Petal.Length'] = pd.DataFrame(pd.cut(df['Petal.Length'], bins=fixed\_bins, labels=['Small', 'Medium', 'Large']))

# Adaptive width binning

df\_adaptive\_binning['Sepal.Width'] = pd.qcut(df['Sepal.Width'], q=5, labels=['Smallest', 'Small', 'Medium', 'Large', 'Largest'])

df\_adaptive\_binning['Petal.Width'] = pd.qcut(df['Petal.Width'], q=5, labels=['Smallest', 'Small', 'Medium', 'Large', 'Largest'])

df\_discretized = pd.concat([df\_fixed\_binning[['Sepal.Length', 'Petal.Length']], df\_adaptive\_binning[['Sepal.Width', 'Petal.Width']], df['Species']], axis=1)

Sepal.Length Petal.Length Sepal.Width Petal.Width Species

0 Medium Small Largest Smallest setosa

1 Medium Small Small Smallest setosa

2 Medium Small Large Smallest setosa

3 Medium Small Medium Smallest setosa

4 Medium Small Largest Smallest setosa

# Plot countplot for 'Sepal.Length'

plt.figure(figsize=(8, 6))

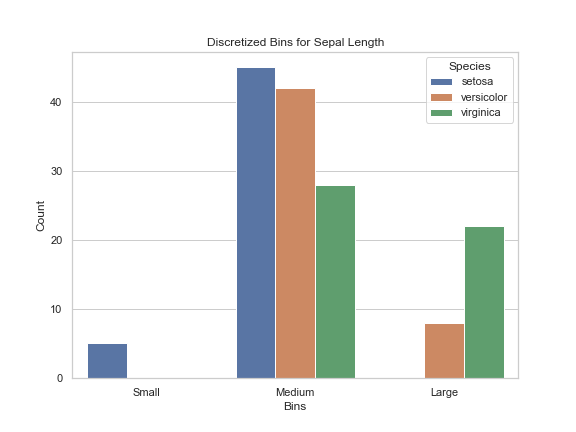
sns.countplot(data=df\_discretized, x='Sepal.Length', hue='Species')

plt.title('Discretized Bins for Sepal Length')

plt.xlabel('Bins')

plt.ylabel('Count')

plt.show()



# Plot countplot for 'Petal.Length'

plt.figure(figsize=(8, 6))

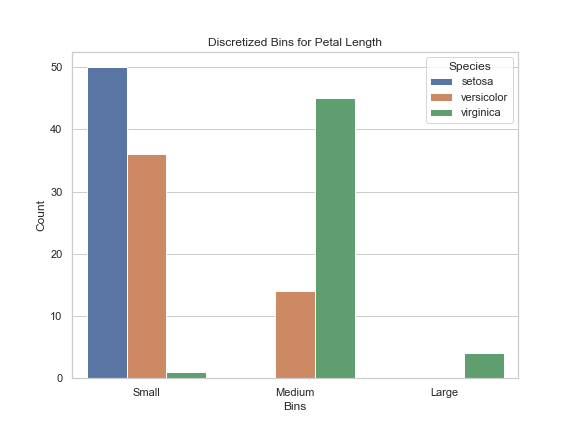
sns.countplot(data=df\_discretized, x='Petal.Length', hue='Species')

plt.title('Discretized Bins for Petal Length')

plt.xlabel('Bins')

plt.ylabel('Count')

plt.show()



# Plot countplot for 'Sepal.Width'

plt.figure(figsize=(8, 6))

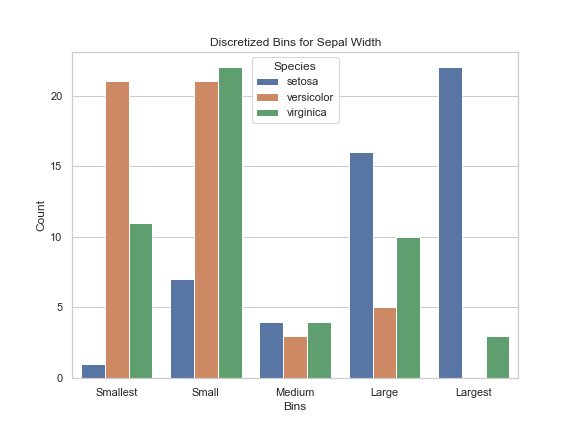
sns.countplot(data=df\_discretized, x='Sepal.Width', hue='Species')

plt.title('Discretized Bins for Sepal Width')

plt.xlabel('Bins')

plt.ylabel('Count')

plt.show()



# Plot countplot for 'Petal.Width'

plt.figure(figsize=(8, 6))

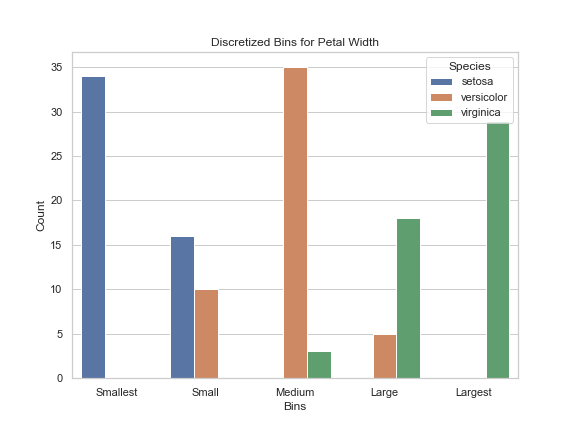
sns.countplot(data=df\_discretized, x='Petal.Width', hue='Species')

plt.title('Discretized Bins for Petal Width')

plt.xlabel('Bins')

plt.ylabel('Count')

plt.show()



|  |  |  |  |
| --- | --- | --- | --- |
| **Name of Feature** | **Description** | **Type** | **Relevance** |
| **ID** |  | **Quantitative/ Nominal** | **Irrelevant (ID does not provide useful information)** |
| Sepal.Length | Length of the sepal in centimeters | Quantitative | Relevant |
| Sepal.Width | Width of the sepal in centimeters | Quantitative | Relevant |
| Petal.Length | Length of the petal in centimeters | Quantitative | Relevant |
| Petal.Width | Width of the petal in centimeters | Quantitative | Relevant |
| Species | Species of iris flower (setosa, versicolor, or virginica) | Nominal | Relevant |