Ex. No. 5

**IMPLEMENTATION OF CLUSTERING ALGORITHM**

**Aim:**

To implement the k-Means Clustering algorithm for the dataset minute weather with scikit-learn in Python.

**Algorithm:**

1. Import necessary libraries:

- pandas

- seaborn

- sklearn

2. Data Loading and Exploration:

a. Read 'minute\_weather.csv' into a DataFrame 'w\_data' selecting columns: 'max\_wind\_direction', 'max\_wind\_speed', 'air\_pressure'.

b. Display the first few rows of 'w\_data'.

3. Data Visualization:

a. Create a scatter plot:

- x-axis: 'max\_wind\_direction'

- y-axis: 'max\_wind\_speed'

- hue: 'air\_pressure'

4. Data Preprocessing:

a. Split the data into:

- Training set: X\_train, Y\_train

- Testing set: X\_test, Y\_test

b. Handle missing values:

- Replace NaN values in X\_train and X\_test with mean of respective columns.

- Replace NaN values in Y\_train and Y\_test with mean of 'air\_pressure'.

c. Normalize the data:

- Normalize X\_train and X\_test.

- Normalize Y\_train and Y\_test.

5. KMeans Clustering:

a. Initialize KMeans with:

- n\_clusters = 3

- random\_state = 0

- n\_init = 'auto'

b. Fit KMeans on normalized X\_train.

6. Visualization of Clusters:

a. Create a scatter plot:

- x-axis: 'max\_wind\_direction'

- y-axis: 'max\_wind\_speed'

- hue: KMeans labels from X\_train.

7. Visualization of Clustered Data and Air Pressure:

a. Create a boxplot:

- x-axis: KMeans labels from X\_train

- y-axis: 'air\_pressure'

8. Additional Visualization (Incomplete/Assumed):

a. Create a scatter plot:

- x-axis: 'max\_wind\_direction'

- y-axis: 'max\_wind\_speed'

- hue: Cluster labels from 'fits[0].labels\_'

**Code:**

import pandas as pd

w\_data = pd.read\_csv('/content/minute\_weather.csv' ,usecols=['max\_wind\_direction','max\_wind\_speed','air\_pressure'])

w\_data.head()

import seaborn as sns

sns.scatterplot(data=w\_data, x='max\_wind\_direction', y='max\_wind\_speed', hue='air\_pressure')

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(w\_data[['max\_wind\_direction','max\_wind\_speed']], w\_data[['air\_pressure']], test\_size=0.33, random\_state=0)

from sklearn import preprocessing

import numpy as np

X\_train\_clean = np.nan\_to\_num(X\_train,nan=np.nanmean(X\_train))

X\_test\_clean = np.nan\_to\_num(X\_test,nan=np.nanmean(X\_test))

X\_train\_norm = preprocessing.normalize(X\_train\_clean)

X\_test\_norm = preprocessing.normalize(X\_test\_clean)

Y\_train\_clean = np.nan\_to\_num(Y\_train,nan=np.nanmean(Y\_train))

Y\_test\_clean = np.nan\_to\_num(Y\_test,nan=np.nanmean(Y\_test))

Y\_train\_norm = preprocessing.normalize(Y\_train\_clean)

Y\_test\_norm = preprocessing.normalize(Y\_test\_clean)

from sklearn.cluster import KMeans

kmeans = KMeans(n\_clusters = 3, random\_state = 0, n\_init='auto')

kmeans.fit(X\_train\_norm)

import seaborn as sns

sns.scatterplot(data = X\_train, x = 'max\_wind\_direction', y='max\_wind\_speed', hue = kmeans.labels\_)

import seaborn as sns

sns.boxplot(x = kmeans.labels\_, y = Y\_train['air\_pressure'])

import seaborn as sns

sns.scatterplot(data = X\_train, x = 'max\_wind\_direction', y = 'max\_wind\_speed', hue = fits[0].labels\_)

**Output:**

air\_pressure min\_wind\_direction min\_wind\_speed

0 912.3 96.0 1.4

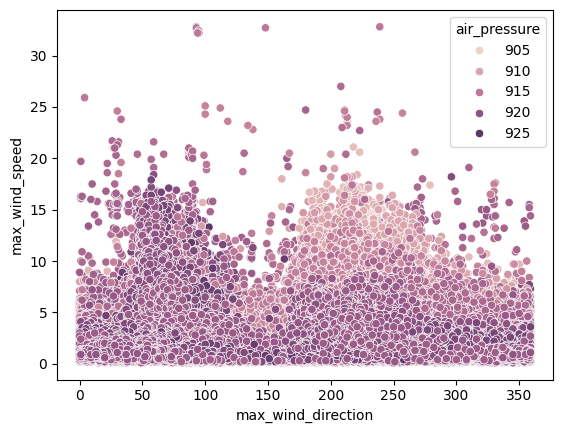
1 912.3 80.0 1.3

2 912.3 70.0 1.2

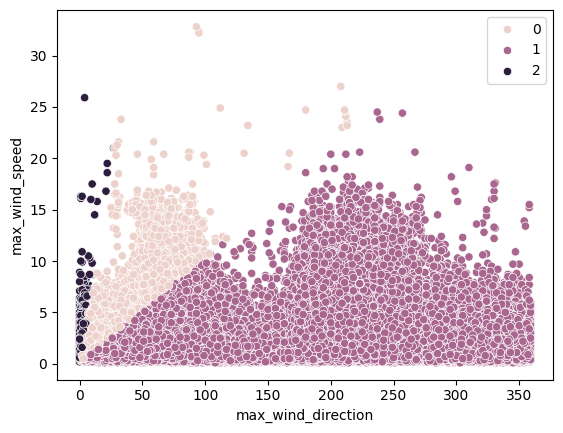
3 912.3 60.0 1.1

4 912.3 30.0 1.0

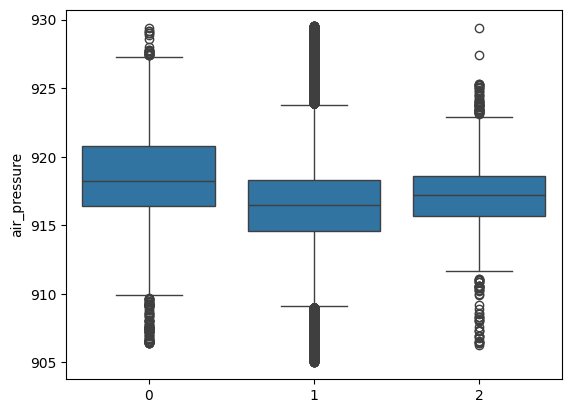
<Axes: xlabel='min\_wind\_direction', ylabel='min\_wind\_speed'>



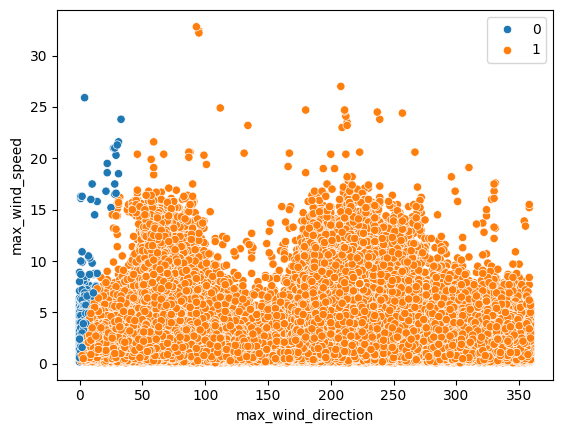
<Axes: xlabel='min\_wind\_direction', ylabel='min\_wind\_speed'>



<Axes: ylabel='air\_pressure'>



<Axes: xlabel='min\_wind\_direction', ylabel='min\_wind\_speed'>



**Rubrics:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Problem**  **Understanding**  **(10)** | **Implementation**  **(20)** | **Viva**  **(10)** | **Time Management**  **(10)** | **Total**  **(50)** |
|  |  |  |  |  |

**Result:**

Thus the implementation of the k-Means Clustering algorithm for the dataset minute weather with scikit-learn in Python was successfully executed and he output was verified.