



# Project Presentation

BHOMIKA  
AMAEDA






# $K$ - $W$ ALGORITHM

ROCK or MINE DATASET



# Introduction

The K-Nearest Neighbors (KNN) algorithm is a simple yet effective supervised machine learning algorithm used for both classification and regression tasks. It operates based on the premise that similar data points are likely to belong to the same class or have similar values.





# Libraries

```
import numpy as np
import pandas as pd
from math import sqrt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
```

# Euclidean distance function

```
def Euclidean_distance(row1, row2):  
    distance = 0  
    for i in range(len(row1)-1):  
        distance += (row1[i] - row2[i])**2 |  $(x1-x2)^2 + (y1-y2)^2$   
    return sqrt(distance)
```

# Load dataset

```
dataset = pd.read_csv('C:\\Users\\Bhumika\\Downloads\\archive (2)\\ROCK_OR_MINE.csv')
```

```
X = dataset.iloc[:, :-1].values # Features  
y = dataset.iloc[:, -1].values  # Target variable
```

test size = 0.2

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
num_neighbors = 5
predicted_labels = knn_predict(X_train, X_test, y_train, num_neighbors)
accuracy = Evaluate(y_test, predicted_labels)
print(f"Accuracy of the KNN model: {accuracy}")
print(f"Predicted labels: {predicted_labels}")
print(f"True labels: {y_test}")
conf_matrix = confusion_matrix(y_test, predicted_labels)
print("Confusion Matrix:")
print(conf_matrix)
```

# Output

```
Accuracy of the KNN model: 0.7380952380952381  
Predicted labels: ['R', 'R', 'R', 'R', 'M', 'R',  
'M', 'M', 'M', 'R', 'M', 'R', 'M', 'M', 'M', 'M']  
True labels: ['M' 'R' 'R' 'M' 'M' 'R' 'M' 'M'  
'R' 'R' 'M' 'M' 'M' 'M' 'R' 'R' 'R' 'R' 'M' 'M'  
'M' 'M' 'R' 'M' 'R' 'M']  
Confusion Matrix:  
[[20  7]  
 [ 4 11]]
```

```
Accuracy of the KNN model: 0.7380952380952381
Predicted labels: ['R', 'R', 'R', 'R', 'M', 'R', 'M', 'M', 'M', 'R', 'M', 'R', 'M', 'M', 'M', 'M', 'R', 'M', 'R', 'R', 'M',
'M', 'M', 'M', 'R', 'M', 'R', 'M', 'M', 'M', 'M', 'R', 'R', 'M', 'R', 'R', 'M', 'R', 'M', 'M', 'R', 'M']
True labels: ['M' 'R' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'M' 'M' 'M' 'M' 'M' 'M'
'R' 'R' 'M' 'M' 'M' 'M' 'R' 'R' 'R' 'R' 'M' 'M' 'M' 'R' 'R' 'M' 'M' 'M'
'M' 'M' 'R' 'M' 'R' 'M']
Confusion Matrix:
[[20  7]
 [ 4 11]]
```



Test size = 0.1

```
X_train2, X_test2, y_train2, y_test2 = train_test_split(X, y, test_size=0.1, random_state=42)
num_neighbors2 = 5
predicted_labels2 = knn_predict(X_train2, X_test2, y_train2, num_neighbors2)
accuracy2 = Evaluate(y_test2, predicted_labels2)
print(f"Accuracy of the KNN model: {accuracy2}")
print(f"Predicted labels: {predicted_labels2}")
print(f"True labels: {y_test2}")
conf_matrix2 = confusion_matrix(y_test2, predicted_labels2)
print("Confusion Matrix:")
print(conf_matrix2)
```

# Output

```
Accuracy of the KNN model: 0.7619047619047619
Predicted labels: ['R', 'R', 'R', 'R', 'M', 'R', 'M', 'M', 'R', 'M', 'R', 'M', 'M', 'M', 'R', 'M', 'R', 'R', 'M']
True labels: ['M' 'R' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'M' 'M' 'M' 'M']
              'R' 'R' 'M']
Confusion Matrix:
[[10  4]
 [ 1  6]]
```

```
Accuracy of the KNN model: 0.7619047619047619
Predicted labels: ['R', 'R', 'R', 'R', 'M', 'R', 'M', 'M', 'M', 'M', 'R', 'M', 'R', 'M', 'M', 'M', 'M', 'R', 'M', 'R', 'R', 'M']
True labels: ['M' 'R' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'R' 'M' 'M' 'M' 'M' 'M' 'M' 'M' 'M']
              'R' 'R' 'M']
Confusion Matrix:
[[10  4]
 [ 1  6]]
```

# Comparison

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

Accuracy: 0.7380952380952381



Thank You

