Experiment:-4

Objective:- KNN

```
import sklearn
import pandas as pd
from sklearn.datasets import load_iris


iris=load_iris()
iris.keys()
df=pd.DataFrame(iris['data'])
df.head()
print(df)
print(iris['target_names'])
iris['feature_names']
```

```
         0    1    2    3
0      5.1  3.5  1.4  0.2
1      4.9  3.0  1.4  0.2
2      4.7  3.2  1.3  0.2
3      4.6  3.1  1.5  0.2
4      5.0  3.6  1.4  0.2
..     ...  ...  ...  ...
145    6.7  3.0  5.2  2.3
146    6.3  2.5  5.0  1.9
147    6.5  3.0  5.2  2.0
148    6.2  3.4  5.4  2.3
149    5.9  3.0  5.1  1.8

[150 rows x 4 columns]
['setosa' 'versicolor' 'virginica']
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

```
X=df
y=iris['target']


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)


from sklearn.neighbors import KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=5)


knn.fit(X_train, y_train)
```

```
    ▾ KNeighborsClassifier
    KNeighborsClassifier()
```

```
from sklearn import metrics
y_pred = knn.predict(X_test)
i = 0
print ("\n-------------------------------------------------------------------------")
print ('%-25s %-25s %-25s' % ('Original Label', 'Predicted Label', 'Correct/Wrong'))
print ("-------------------------------------------------------------------------")
for label in y_test:
    print ('%-25s %-25s' % (label, y_pred[i]), end="")
    if (label == y_pred[i]):
        print (' %-25s' % ('Correct'))
    else:
        print (' %-25s' % ('Wrong'))
    i = i + 1
print ("-------------------------------------------------------------------------")
print("\nConfusion Matrix:\n",metrics.confusion_matrix(y_test, y_pred))
print ("-------------------------------------------------------------------------")
print("\nClassification Report:\n",metrics.classification_report(y_test, y_pred))
```

```
print ("------------------------------------------------------------------------")
print('Accuracy of the classifer is %0.2f' % metrics.accuracy_score(y_test,y_pred))
print ("------------------------------------------------------------------------")
```

```
       -------------------------------------------------------------------
       Original Label          Predicted Label        Correct/Wrong
       -------------------------------------------------------------------
       1                       1                      Correct
       0                       0                      Correct
       2                       2                      Correct
       1                       1                      Correct
       1                       1                      Correct
       0                       0                      Correct
       1                       1                      Correct
       2                       2                      Correct
       1                       1                      Correct
       1                       1                      Correct
       2                       2                      Correct
       0                       0                      Correct
       0                       0                      Correct
       0                       0                      Correct
       0                       0                      Correct
       1                       1                      Correct
       2                       2                      Correct
       1                       1                      Correct
       1                       1                      Correct
       2                       2                      Correct
       0                       0                      Correct
       2                       2                      Correct
       0                       0                      Correct
       2                       2                      Correct
       2                       2                      Correct
       2                       2                      Correct
       2                       2                      Correct
       2                       2                      Correct
       0                       0                      Correct
       0                       0                      Correct
       0                       0                      Correct
       0                       0                      Correct
       1                       1                      Correct
       0                       0                      Correct
       0                       0                      Correct
       2                       2                      Correct
       1                       1                      Correct
       0                       0                      Correct
       0                       0                      Correct
       0                       0                      Correct
       2                       2                      Correct
       1                       1                      Correct
       1                       1                      Correct
       0                       0                      Correct
       0                       0                      Correct
       1                       1                      Correct
       2                       1                      Wrong
       2                       2                      Correct
       1                       1                      Correct
       2                       2                      Correct
       -------------------------------------------------------------------

       Confusion Matrix:
        [[19  0  0]
```

```
seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

```
     86400
```

```
seconds_in_a_week = 7 * seconds_in_a_day
seconds_in_a_week
```
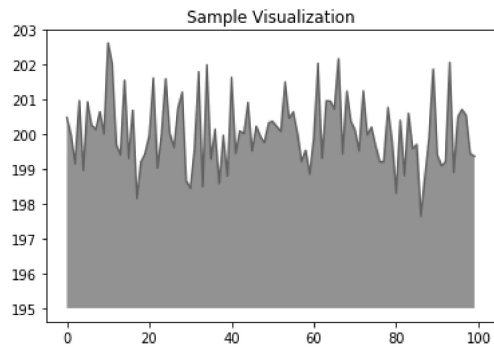
```
     604800
```

```
import numpy as np
from matplotlib import pyplot as plt

ys = 200 + np.random.randn(100)
x = [x for x in range(len(ys))]

plt.plot(x, ys, '-')
plt.fill_between(x, ys, 195, where=(ys > 195), facecolor='g', alpha=0.6)
```

```
plt.title("Sample Visualization")
plt.show()
```


Sample Visualization