Experiment-8:K Nearest neighbours algorithm to classify iris data set

```
import sklearn
import pandas as pd
from sklearn.datasets import load_iris
iris=load_iris()
iris.keys()
df=pd.DataFrame(iris['data'])
print(df)
print(iris['target_names'])
iris['feature_names']
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=3)
knn.fit(X_train,y_train)
import numpy as np
x_new=np.array([[5,2.9,1,0.2]])
prediction=knn.predict(x_new)
iris['target_names'][prediction]
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
y_pred=knn.predict(X_test)
cm=confusion_matrix(y_test,y_pred)
print(cm)
print(" correct predicition",accuracy_score(y_test,y_pred))
print(" worng predicition",(1-accuracy_score(y_test,y_pred)))
```

Output:

```
       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']
[[19  0  0]
 [ 0 15  0]
 [ 0  1 15]]
 correct predicition 0.98
 worng predicition 0.020000000000000018
```

Experiment -9:Decision Tree Classifier

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
# Load the Iris dataset
iris = load_iris()
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.2,
random_state=42)
# Create a decision tree classifier
clf = DecisionTreeClassifier()
# Train the classifier on the training data
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
#Evaluate the accuracy of the model-
accuracy = accuracy_score(y_test, y_pred)
sepallength=float(input("ENTER SEPAL LENGTH OF FLOWER:"))
sepalwidth=float(input("ENTER SEPAL WIDTH OF FLOWER:"))
petallength=float(input("ENTER PETAL LENGTH OF FLOWER:"))
petalwidth=float(input("ENTER PETAL WIDTH OF FLOWER:"))
print("Accuracy:", accuracy)
new_sample = [[sepallength,sepalwidth,petallength,petalwidth]]
predicted_class = clf.predict(new_sample)
predicted_species = iris.target_names[predicted_class]
print("Predicted species:", predicted_species)
```

Output:

```
ENTER SEPAL LENGTH OF FLOWER:5
ENTER SEPAL WIDTH OF FLOWER:5
ENTER PETAL LENGTH OF FLOWER:6
ENTER PETAL WIDTH OF FLOWER:4
Accuracy: 1.0
Predicted species: ['virginica']
```