7. Implement and demonstrate the working of k-Nearest Neighbour Algorithm and apply it to classify the iris data set.

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import warnings

# Load Iris dataset
iris = load_iris()

# Column names
column_names = iris.feature_names
print("Column names in the Iris dataset:")
print(column_names)

# Target names
target_names = iris.target_names
print("\nTarget names in the Iris dataset:")
print(target_names)

# Create a DataFrame
iris_df = pd.DataFrame(data=iris['data'],
columns=iris['feature_names'])
print(iris_df.head())
iris_df['target'] = iris['target']


# Split the dataset into features and target
X = iris_df.drop('target', axis=1)
y = iris_df['target']

# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Instantiate the k-Nearest Neighbor classifier
k = 3
knn = KNeighborsClassifier(n_neighbors=k)

# Fit the classifier to the training data
knn.fit(X_train, y_train)

# Predict the labels for the test set
y_pred = knn.predict(X_test)
```

```
# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy of k-Nearest Neighbor (k={k}): {accuracy}")

# Display actual and predicted outputs
actual_predicted_df = pd.DataFrame({'Actual': y_test, 'Predicted':
y_pred})
print(actual_predicted_df)

# Demo: Classify a sample
sample = np.array([[5.1, 3.5, 1.4, 0.2]])  # Example sample
predicted_class = knn.predict(sample)[0]  # Extract scalar value
predicted_species = iris.target_names[predicted_class]
print(f"Predicted class for the sample: {predicted_species}")

# Suppress the feature names warning
warnings.filterwarnings("ignore", message="X does not have valid
feature names")
```