auto-water-supply-merge

June 28, 2024

[ ]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import mean_squared_error, r2_score

from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier
```

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:
```
data = pd.read_csv('/content/drive/MyDrive/PUMPING/data.csv')
```

[ ]:
```
x= [['moisture'],['temp']]
y = ['pump']
```

[ ]:
```
# Assuming 'data' is a pandas DataFrame loaded from your CSV file
x = data[['moisture', 'temp']]
y = data['pump']

LR = LinearRegression()
LR.fit(x, y)
```

[ ]:
```
LinearRegression()
```

[ ]:
```
LO= LogisticRegression()
LO.fit(x,y)
```

[ ]:
```
LogisticRegression()
```

[ ]:
```
k=3
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(x, y)
```

[ ]:
```
KNeighborsClassifier(n_neighbors=3)
```

[ ]:
```
DTC = DecisionTreeClassifier()
DTC.fit(x,y)
```

[ ]:
```
DecisionTreeClassifier()
```

[ ]:
```
RFC = RandomForestClassifier(random_state=0)
RFC.fit(x,y)
```

[ ]:
```
RandomForestClassifier(random_state=0)
```

[ ]:
```
# Get user input for moisture and temperature
user_moisture = float(input("Enter moisture value: "))
user_temp = float(input("Enter temperature value: "))

# Create a DataFrame for user input
user_input = pd.DataFrame({'moisture': [user_moisture], 'temp': [user_temp]})

# Make the prediction
prediction = LR.predict(user_input)


print("Predicted pump value by linearR:", prediction[0])

if prediction[0] < 0.5:
    print("Pump is off")
else:
    print("Pump is on")

prediction = LO.predict(user_input)

print("Predicted pump value by logisticR:", prediction[0])

if prediction[0] < 0.5:
    print("Pump is off")
else:
    print("Pump is on")

prediction = knn.predict(user_input)
```

```
print("Predicted pump value by KNN:", prediction[0])

if prediction[0] < 0.5:
    print("Pump is off")
else:
    print("Pump is on")

user_pred = DTC.predict(user_input)

print("Predicted pump value by DTree:", user_pred[0])

if user_pred[0] < 0.5:
    print("Pump is off")
else:
    print("Pump is on")

user_pred = RFC.predict(user_input)

print("Predicted pump value by RForest:", user_pred[0])

if user_pred[0] < 0.5:
    print("Pump is off")
else:
    print("Pump is on")
```

```
Enter moisture value: 638
Enter temperature value: 16
Predicted pump value by linearR: 0.7473335801680322
Pump is on
Predicted pump value by logisticR: 1
Pump is on
Predicted pump value by KNN: 1
Pump is on
Predicted pump value by DTree: 1
Pump is on
Predicted pump value by RForest: 1
Pump is on
```

[ ]: