**Smart Agriculture Monitoring**

|  |  |
| --- | --- |
| **Sr No.** | **Aim of Module** |
| **1.** | **Applying Machine Learning to predict data on Device Level**  **THEORY:** Decision tree classifiers are utilized as a well-known classification technique in different pattern recognition issues, for example, image classification and character recognition (Safavian & Landgrebe, 1991). Decision tree classifiers perform more successfully, specifically for complex classification problems, due to their high adaptability and computationally effective features. Besides, decision tree classifiers exceed expectations over numerous typical supervised classification methods (Friedl & Brodley, 1997).  In particular, no distribution assumption is needed by decision tree classifiers regarding the input data. This particular feature gives to the Decision Tree Classifiers a higher adaptability to deal with different datasets, whether numeric or categorical, even with missing data. Also, decision tree classifiers are basically nonparametric. Also, decision trees are ideal for dealing with nonlinear relations among features and classes. At long last, the classification procedure through a tree-like structure is constantly natural and interpretable.  **PROGRAM CODE:**  import pandas as pd # data processing, CSV file I/O  from sklearn.model\_selection import train\_test\_split  from sklearn.tree import DecisionTreeClassifier  import random as rn  data= pd.read\_csv('1119492.csv') #importing CSV file  data.drop(["d\_id",],axis=1,inplace=True) #dropping columns that are not required  data.drop(["time\_stamp",],axis=1,inplace=True)  data.drop(["ldr"],axis=1,inplace=True)  data['hum']=data['hum'].astype('float32') #converting datatype to type supported by Classifier  data['soil']=data['soil'].astype('float32')  data['temp']=data['temp'].astype('float32')  data['pump']=data['pump'].astype('float32')  #data.info()  y=data.pump.values #assigning values in column 'pump' as labels  #print(y)  data.drop(["pump"],axis=1,inplace=True) #dropping pump as it is a label  x\_data=data #assigning rest of data as input features  #print(x\_data)  x\_train,x\_test,y\_train,y\_test=train\_test\_split(x\_data,y,test\_size=0.5,random\_state=10) #splitting data into training and testing data also randomizing  #print(x\_train)  #print(x\_test)  dt= DecisionTreeClassifier() #Initialzing Classifier  dt.fit(x\_train,y\_train) #Training the Classifier  print("Score : ",dt.score(x\_test,y\_test)) #Testing the Classifier  prediction=[None]\*10 #Declaring list to hold predictions of 10 rows of randomly generated data  for i in range(len(prediction)):  prediction[i]= dt.predict([[rn.randrange(50),rn.randrange(60),rn.randrange(100)]])  print(prediction[i][0])  **OUTPUT**: |