

Here in the first model there are 8 parameters considered as indicated in the column names , each with their min max values , the data entered should be within or bounded within these values

the avg temp & rainfall of current year should be forecasted which I guess there are some other api links which provide for the current location .

ph, n,p,k will be provided manually,

present day temp will be taken as sowing temp or given as preferred by user,

but harvesting temp must be calculated by some method . ( take a look at this code and figure out how to implement them)

<https://github.com/shan515/AI-Based_Crop-Predictor_App/blob/main/Server/app.py>

the model is 98-100 % accurate on any valid data.

These is a below sample how our inputs should be.

Try these codes its optional , ill explain if we meet.

#Loading and testing

import pickle

from sklearn.preprocessing import StandardScaler

try these code .

#for the first model for here I use forest classifer.

loaded\_model=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/southside/forest\_classifier.pkl','rb'))

loaded\_scale=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/southside/scale.pkl','rb'))

data =loaded\_scale.transform( np.array([[24.115815, 8.600399, 7.745665, 23.796676,  21.889624,1000, 42.592624,  0.000000  ]]))

prediction = loaded\_model.predict(data)

print(prediction)

#['coriander']

#using neural network model

import pickle

from sklearn.preprocessing import StandardScaler

import numpy as np

loaded\_model=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/southside/ann.pkl','rb'))

loaded\_scale=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/southside/scale.pkl','rb'))

data =loaded\_scale.transform( np.array([[ 25.499573,  7.405498  ,298.897352 ,30.325860  ,21.564118, 9.228892, 10.670175,  10.100367 ]]))

prediction = loaded\_model.predict(data)[0]

top3\_indices = np.argsort(prediction)[-3:]

top3\_codes = [get\_key(idx) for idx in top3\_indices]

top3\_predictions = [prediction[idx] for idx in top3\_indices]

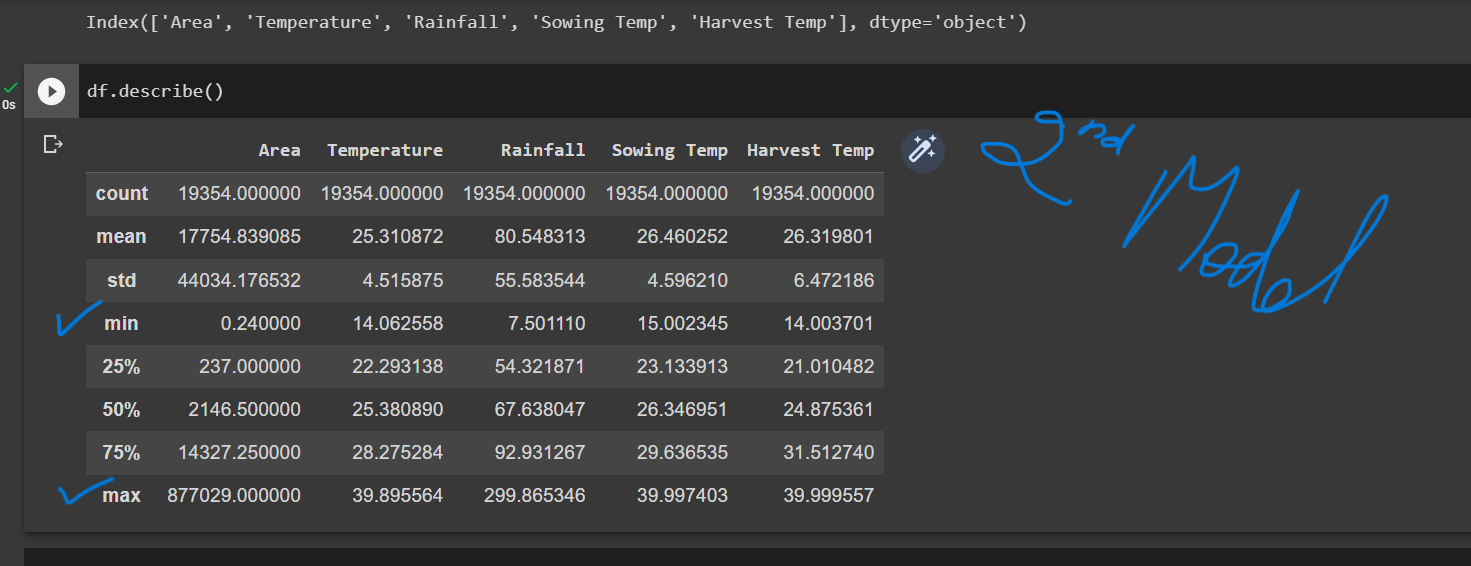
print("Top 3 codes:", top3\_codes)

print("Top 3 predictions:", top3\_predictions)

code = np.argmax(prediction)

accuracy = np.max(prediction) \* 100

print("Code is", code, "with accuracy", accuracy, "%")



Once the crop is predicted we have to take it and append corresponding production pickle file ;

Example. If onion was recommended , then we traverse the production directory for onion pickle file in the backend and provide further inputs.

(ie for each crop there are different pickle files for production).

For the second model we consider 5 features some might be repeatative but important ; again here to the values entered must be bounded to min and max

Annual temp , rainfall, sowing temp, and harvesting temp can be done as did in first model.

Area can be anything and can be done manually.

( suggestion : try to include area and hide other features buttons or take automatically from the data entered previously in the first model).

Here model is 75% accurate on any valid data , though there are exceptions.

these is how the sample input should be.

loaded\_model=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/crop\_yield\_kar/crop\_wheat.csv.pkl','rb'))

loaded\_scale=pickle.load(open('/content/drive/MyDrive/github/collegeproject/pickle\_files/scalers\_for\_crop/crop\_wheat.csv\_scaler.pkl','rb'))

inp=[ 3566  ]

data =loaded\_scale.transform([inp])

prediction = loaded\_model.predict(data)

y\_new\_inverse = loaded\_scale.inverse\_transform([prediction])

print(y\_new\_inverse)