|  |
| --- |
| from sklearn.model\_selection import train\_test\_split |
|  | import pandas as pd |
|  | from sklearn.svm import SVC |
|  | from sklearn.naive\_bayes import GaussianNB |
|  | from sklearn.ensemble import RandomForestClassifier, VotingClassifier |
|  | from sklearn.neighbors import KNeighborsClassifier |
|  | from sklearn.metrics import accuracy\_score |
|  | from sklearn import model\_selection |
|  |  |
|  | crop = pd.read\_csv('Data/crop\_recommendation.csv') |
|  | X = crop.iloc[:,:-1].values |
|  | Y = crop.iloc[:,-1].values |
|  |  |
|  | X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size = 0.15) |
|  |  |
|  | models = [] |
|  | models.append(('SVC', SVC(gamma ='auto', probability = True))) |
|  | models.append(('svm1', SVC(probability=True, kernel='poly', degree=1))) |
|  | models.append(('svm2', SVC(probability=True, kernel='poly', degree=2))) |
|  | models.append(('svm3', SVC(probability=True, kernel='poly', degree=3))) |
|  | models.append(('svm4', SVC(probability=True, kernel='poly', degree=4))) |
|  | models.append(('svm5', SVC(probability=True, kernel='poly', degree=5))) |
|  | models.append(('rf',RandomForestClassifier(n\_estimators = 21))) |
|  | models.append(('gnb',GaussianNB())) |
|  | models.append(('knn1', KNeighborsClassifier(n\_neighbors=1))) |
|  | models.append(('knn3', KNeighborsClassifier(n\_neighbors=3))) |
|  | models.append(('knn5', KNeighborsClassifier(n\_neighbors=5))) |
|  | models.append(('knn7', KNeighborsClassifier(n\_neighbors=7))) |
|  | models.append(('knn9', KNeighborsClassifier(n\_neighbors=9))) |
|  |  |
|  | vot\_soft = VotingClassifier(estimators=models, voting='soft') |
|  | vot\_soft.fit(X\_train, y\_train) |
|  | y\_pred = vot\_soft.predict(X\_test) |
|  |  |
|  | scores = model\_selection.cross\_val\_score(vot\_soft, X\_test, y\_test,cv=5,scoring='accuracy') |
|  | print("Accuracy: ",scores.mean()) |
|  |  |
|  | score = accuracy\_score(y\_test, y\_pred) |
|  | print("Voting Score % d" % score) |
|  |  |
|  | import pickle |
|  | pkl\_filename = 'Crop\_Recommendation.pkl' |
|  | Model\_pkl = open(pkl\_filename, 'wb') |
|  | pickle.dump(vot\_soft, Model\_pkl) |
|  | Model\_pkl.close() |