Title: Mushroom classification

Code:

import numpy as np *# linear algebra*

import pandas as pd

import os

for dirname, \_, filenames in os.walk('/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))

mushroom = pd.read\_csv('mushrooms.csv')

mushroom.info()

mushroom.head(5).transpose()

mushroom.isna().sum()

mushroom.duplicated().sum()

for i in mushroom.columns:

    print(mushroom[**str**(i)].value\_counts())

    print('\n')

mushroom\_en = pd.get\_dummies(mushroom, drop\_first = True)

mushroom\_en.shape

mushroom\_en.columns

X\_feature = **list**(mushroom\_en.columns)

X\_feature.remove('class\_p')

X = mushroom\_en[X\_feature]

X.columns

Y = mushroom\_en['class\_p']

Y.head()

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,Y, test\_size = 0.30, random\_state = 42)

print("X\_train.shape",X\_train.shape)

print("X\_test.shape",X\_test.shape)

print("y\_train.shape",y\_train.shape)

print("y\_test.shape",y\_test.shape)

from sklearn.neighbors import **KNeighborsClassifier**

knn = **KNeighborsClassifier**(n\_neighbors = 21)

knn.fit(X\_train, y\_train)

*#predicting for test data*

knn\_test\_pred = knn.predict(X\_test)

*#evaluate the model*

from sklearn.metrics import accuracy\_score

print(accuracy\_score(y\_test, knn\_test\_pred))

pd.crosstab(y\_test, knn\_test\_pred, rownames = ['Actual'], colnames= ['Predictions'])

*#error on train data*

knn\_train\_pred = knn.predict(X\_train)

print(accuracy\_score(y\_train, knn\_train\_pred))

pd.crosstab(y\_train, knn\_train\_pred, rownames = ['Actual'], colnames = ['Predictions'])

acc = []

*# running KNN algorithm for 3 to 50 nearest neighbours(odd numbers) and*

*# storing the accuracy values*

for i in **range**(3,50,2):

    neigh = **KNeighborsClassifier**(n\_neighbors=i)

    neigh.fit(X\_train, y\_train)

    train\_acc = np.mean(neigh.predict(X\_train) == y\_train)

    test\_acc = np.mean(neigh.predict(X\_test) == y\_test)

    acc.append([train\_acc, test\_acc])

import matplotlib.pyplot as plt *# library to do visualizations*

*# train accuracy plot*

plt.plot(np.arange(3,50,2),[i[0] for i in acc],"ro-")

Output:



