import pandas as pd

import numpy as np

df = pd.read\_csv('/content/sample\_data/pulsar\_stars\_new (1).csv')

df.head

df.shape

#ANswer 1 shape[0]

df = df[((df['TG'] == 0) & (df['MIP'] >= 102.234375) & (df['MIP'] <= 102.6015625)|

        (df['TG'] == 1 ) & (df['MIP'] >= 77.4921875) & (df['MIP'] <= 83.7734375))]

df.shape

answer 2

df['MIP'].mean()

from sklearn.preprocessing import MinMaxScaler

X\_train = df.drop('TG',axis = 1) # delete colum TG

y\_train = df['TG'].values

print(X\_train)

print(X\_train.shape)

scaler = MinMaxScaler().fit(X\_train, y\_train)

X\_train\_scaled = scaler.transform(X\_train)

print(X\_train\_scaled)

print(X\_train\_scaled.shape)

#Answer 03

print(df.keys())

data = pd.DataFrame(data = X\_train\_scaled, columns = df.keys().drop(['TG']))

data.describe()

data.head()

data['MIP'].mean()

from sklearn.linear\_model import LogisticRegression

reg = LogisticRegression(random\_state = 2019, solver = 'lbfgs')

reg.fit(X\_train\_scaled, y\_train)

#Answer 4

new\_obj = [[0.237, 0.293, 0.897, 0.0, 0.841, 0.969, 0.138, 0.083]]

reg.predict\_proba(new\_obj)

from sklearn.neighbors import KNeighborsClassifier

neigh = KNeighborsClassifier(n\_neighbors= 1, p = 2).fit(X\_train\_scaled, y\_train)

#Answer 5

neigh.kneighbors(new\_obj)

from keras.datasets import mnist

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

x\_train = x\_train.reshape(60000,28\*28)

from sklearn.decomposition import PCA

import numpy as np

pca = PCA(svd\_solver='full').fit(x\_train)

exp\_var = np.cumsum(pca.explained\_variance\_ratio\_)

i = 0

for val in exp\_var:

  i += 1;

  if (val > 0.82):

    break

print(i)

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

pca = PCA(n\_components=49, svd\_solver='full').fit(x\_train)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_train, y\_train, test\_size=0.3, random\_state=68)

x\_train = pca.transform(x\_train)

x\_test = pca.transform(x\_test)

np.mean(x\_train[:,0])

from sklearn.multiclass import OneVsRestClassifier

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(criterion = 'gini', min\_samples\_leaf=10, max\_depth=20, n\_estimators=10, random\_state=68)

ovr\_rfc = OneVsRestClassifier(rfc).fit(x\_train, y\_train)

y\_pred = ovr\_rfc.predict(x\_test)

np.sum((y\_pred == y\_test)&(y\_test == 3))

from sklearn.linear\_model import LogisticRegression

lr = LogisticRegression(solver='lbfgs', random\_state=68)

ovr\_lr = OneVsRestClassifier(lr).fit(x\_train, y\_train)

y\_pred = ovr\_lr.predict(x\_test)

np.sum((y\_pred == y\_test)&(y\_test == 8))

from sklearn.tree import DecisionTreeClassifier

dtc = DecisionTreeClassifier(criterion='gini', min\_samples\_leaf=10, max\_depth=20, random\_state=68)

ovr\_dtc = OneVsRestClassifier(dtc).fit(x\_train, y\_train)

y\_pred = ovr\_dtc.predict(x\_test)

np.sum((y\_pred == y\_test)&(y\_test == 5))

import pandas as pd

df = pd.read\_csv('pred\_for\_task.csv', index\_col='FileName')

x\_file = df.drop('Label', axis = 1)

x\_file = pca.transform(x\_file)

y\_pred1 = ovr\_rfc.predict\_proba(x\_file)

print(np.max(y\_pred1[17]))

y\_pred2 = ovr\_lr.predict\_proba(x\_file)

np.max(y\_pred2[14])

y\_pred3 = ovr\_dtc.predict\_proba(x\_file)

np.max(y\_pred3[18])