Name: Praveen kumar K

Reg no: RA1911030010069

EXPERIMENT 12: Development of ensemble model for an application

**AIM:**

To develop a model an ensemble model for an application using python.

**Theory:**

Ensemble learning helps improve machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model. Basic idea is to learn a set of classifiers (experts) and to allow them to vote.

**Algorithm:**

* Import the required libraries.
* Import the data file in the program.
* Clean the data.
* Find the required features on which the model predicts.
* Split the data into two parts say X and Y.
* Train the model using the data X and find the mean absolute error with the predicted value and Y.
* We use random forest and decision trees algorithms to predict our model.
* A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging.

**CODE:**

1. Bagged decision trees:

import pandas

from sklearn import model\_selection

from sklearn.ensemble import BaggingClassifier

from sklearn.tree import DecisionTreeClassifier

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age','class']

dataframe = pandas.read\_csv(url, names=names)

array = dataframe.values

X = array[:,0:8]

Y = array[:,8]

seed = 7

kfold = model\_selection.KFold(n\_splits=10, random\_state=seed, shuffle=True)

cart = DecisionTreeClassifier()

num\_trees = 100

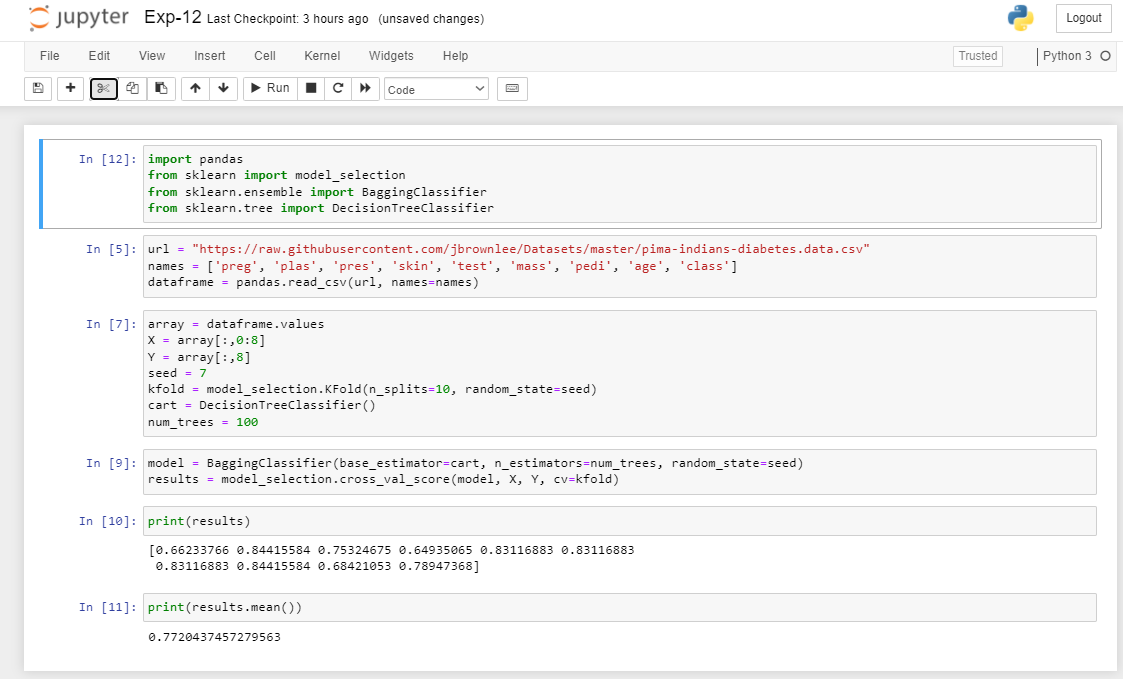
model = BaggingClassifier(base\_estimator=cart, n\_estimators=num\_trees, random\_state=seed)

results = model\_selection.cross\_val\_score(model, X, Y, cv=kfold)

print(results)

print(results.mean())

OUTPUT:



1. Random Forest:

import pandas

from sklearn import model\_selection

from sklearn.ensemble import RandomForestClassifier

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age','class']

dataframe = pandas.read\_csv(url, names=names)

array = dataframe.values

X = array[:,0:8]

Y = array[:,8]

seed = 7

num\_trees = 100

max\_features = 3

kfold = model\_selection.KFold(n\_splits=10, random\_state=seed, shuffle=True)

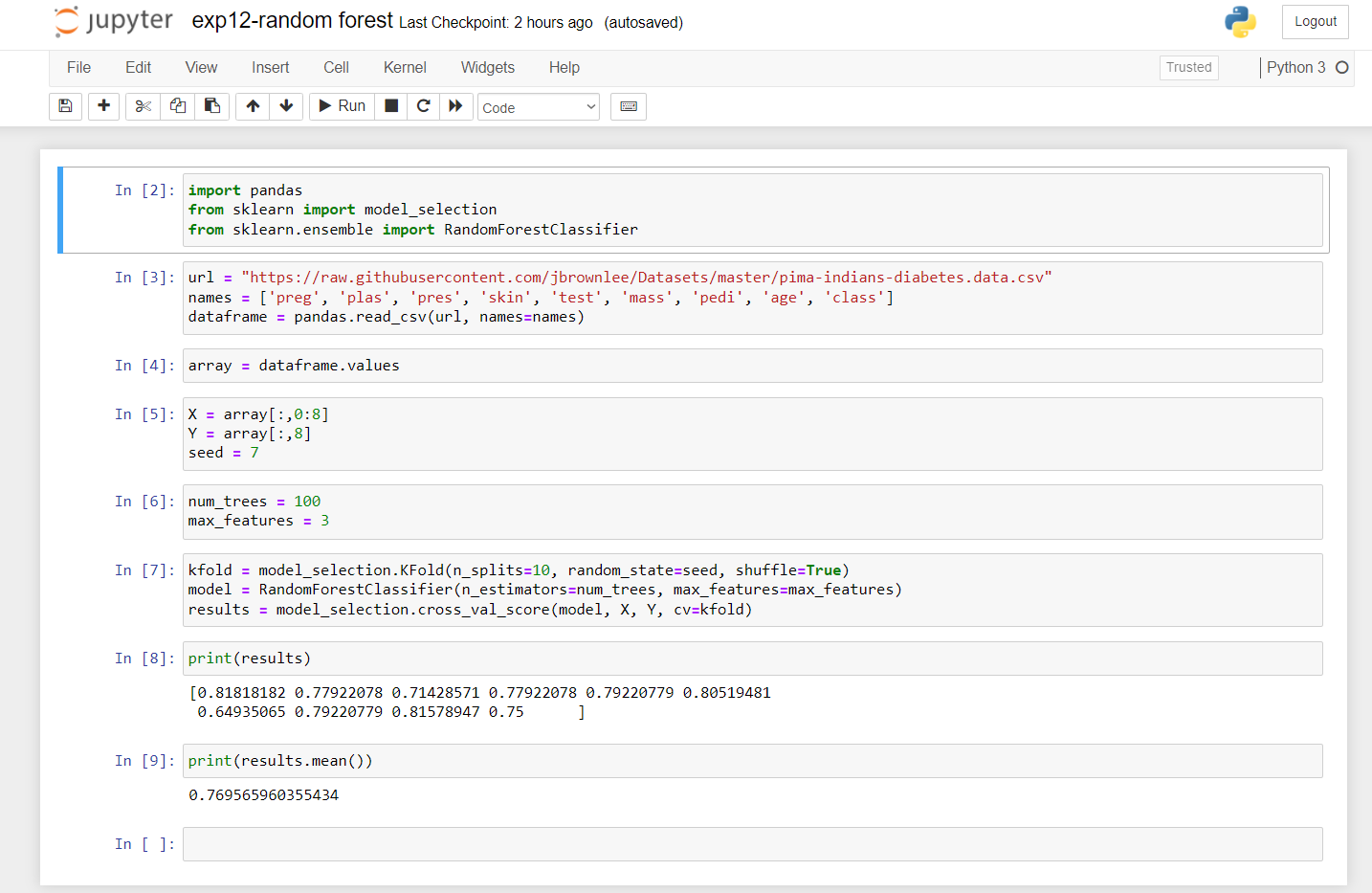
model = RandomForestClassifier(n\_estimators=num\_trees, max\_features=max\_features)

results1 = model\_selection.cross\_val\_score(model, X, Y, cv=kfold)

print(results1)

print(results1.mean())

OUTPUT:



1. Extra trees:

import pandas

from sklearn import model\_selection

from sklearn.ensemble import ExtraTreesClassifier

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"

names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age','class']

dataframe = pandas.read\_csv(url, names=names)

array = dataframe.values

X = array[:,0:8]

Y = array[:,8]

seed = 7

num\_trees = 100

max\_features = 7

kfold = model\_selection.KFold(n\_splits=10, random\_state=seed, shuffle=True)

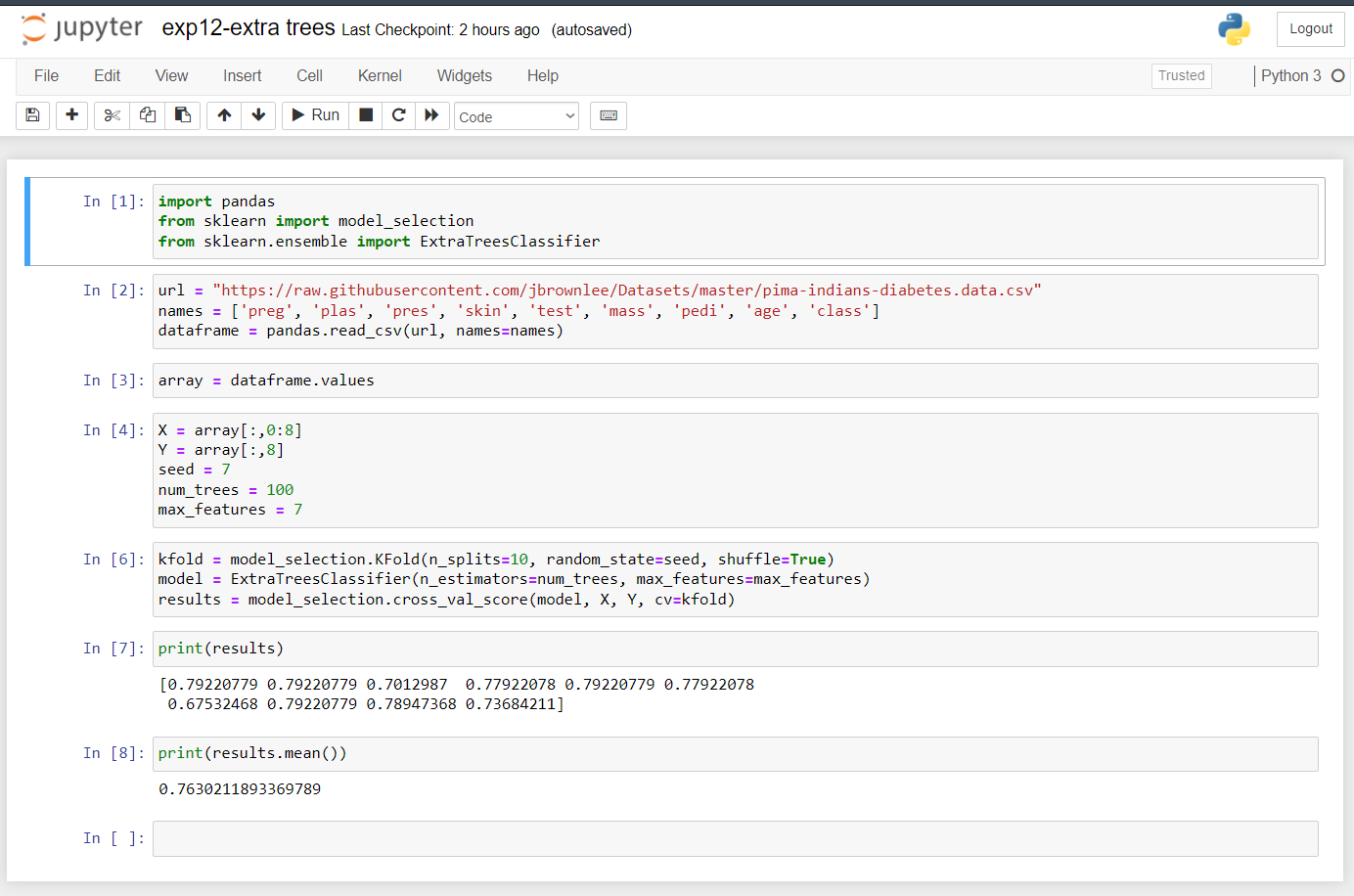
model = ExtraTreesClassifier(n\_estimators=num\_trees, max\_features=max\_features)

results2 = model\_selection.cross\_val\_score(model, X, Y, cv=kfold)

print(results2)

print(results2.mean())

OUTPUT:



Verification:

From both the algorithms we can see the difference in the mean absolute error. The predictions made are nearby the data which are input, so we can verify the data is correctly predicted as it lies along the dataset provided.

RESULT: Hence, successfully implemented the problem and verified the output and document result.