DATA7202 : Assessment 3

Name : Peng Yu

Student ID : 46635884

Question 1

Answer:

Decision tree leads to different results according to different  and , so the decision tree model is easy to overfit. Assume that when the first node is , the result is ; if not, the depth of the tree is increased, and determine whether to , if it is, the result is , if not, to increase the depth of the tree again, and so on., when the depth of the decision tree is equal to the sample size of the training set, any training set can be fitted via a tree with zero training loss.

Question 2

Answer:

Suppose there is a c that minimizes the squared-error loss.

Then

The transformed expression can be equivalent to a quadratic equation of c of one variable, which is the minimum value for .

Therefore is minimizes the squared-error loss.

Question 3

Answer:

First there are 3 blue and 2 red data points in a certain tree region.

Then , .

Misclassification impurity:

Gini impurity:

Entropy impurity:

Second for 2 blue and 3 red data points.

Misclassification impurity:

Gini impurity:

Entropy impurity:

In summary, the results of “3 blue and 2 red” and “2 blue and 3 red” are the same.

Question 4

Answer:

Suppose p is that contains a fraction of the points from

When n = 2,

When n = 3,

When n = 4,

When n = 5,

Therefore when n = n

Therefore, (1 - p) is that does not contain a fraction of the points from

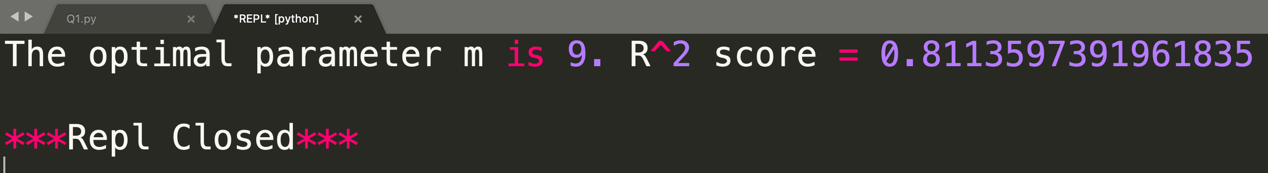
When n is large

From what has been discussed above that for large n, does not contain a fraction of about of the points from

Question 5

Answer:

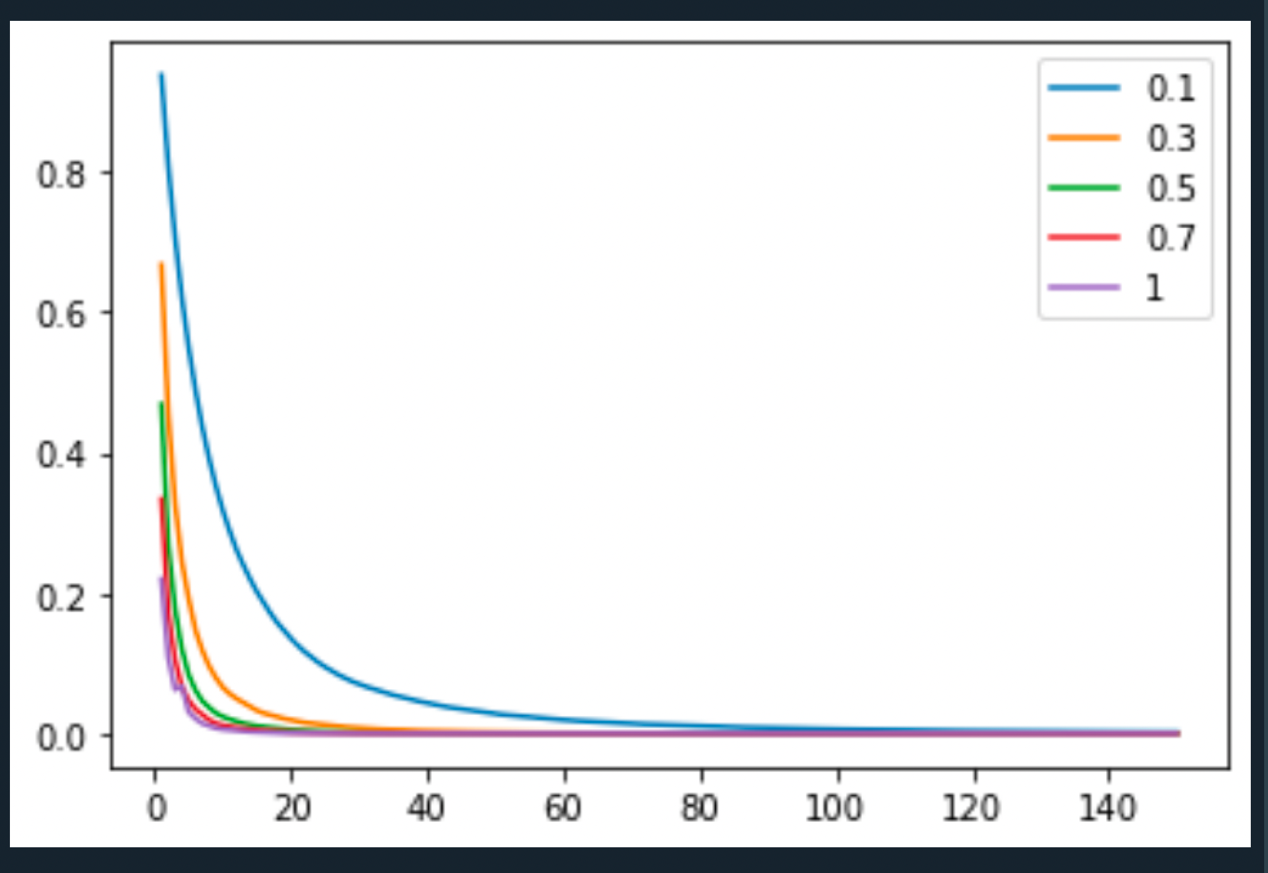
The optimal parameter m is 9. R^2 score = 0.8113597391961835.



Question 6

Answer:

As can be seen from the figure below, the smaller a is, the faster the slope of decreases with the increase of B. When the range of B is 0 to 20, the curve drops the fastest. When B is greater than 100, all the curves approach 0.



**Code Appendix**

**Question 5**

import numpy as np

from sklearn.datasets import make\_friedman1

from sklearn.ensemble import RandomForestRegressor

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

from sklearn.metrics import r2\_score

# create regression problem

n\_points = 1000 # points

scores=[]

x, y = make\_friedman1(n\_samples=n\_points, n\_features=15,

noise=1.0, random\_state=100)

# split to train/test set

x\_train, x\_test, y\_train, y\_test = \

train\_test\_split(x, y, test\_size=0.33, random\_state=100)

#Answer:

for i in np.arange(1,15):

rf = RandomForestRegressor(n\_estimators=1000, oob\_score = True, max\_features=i, random\_state=100)

rf.fit(x\_train,y\_train)

yhatrf = rf.predict(x\_test)

scores.append(r2\_score(y\_test, yhatrf))

print('m='+str(i)+'. r2='+str(scores[i-1]))

print('The optimal parameter m is '+ str(scores.index(max(scores))+1) + ". R^2 score = "+ str(max(scores)))

**Question 6**

from sklearn.datasets import make\_blobs

from sklearn.metrics import zero\_one\_loss

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

import numpy as np

import matplotlib.pyplot as plt

from sklearn.ensemble import GradientBoostingClassifier

X\_train, y\_train = make\_blobs(n\_samples=1000, n\_features=10, centers=3, random\_state=10, cluster\_std=5)

#Answer:

gammas = [0.1, 0.3, 0.5, 0.7, 1]

for gamma in gammas:

GBC = GradientBoostingClassifier(learning\_rate=gamma, n\_estimators = 150)

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

plt.plot(np.arange(1,151,1), GBC.train\_score\_, label = gamma)

plt.legend()

plt.show()