ECON 425 HW4 Solutions

January 30, 2024

Problem 1 (suboptimality of ID3)

For this problem, use pen and paper only. Let $\mathcal{X} = \{0,1\}^3$, $\mathcal{Y} = \{0,1\}$ and consider the training set

$$\{(x_i, y_i)\}_{i=1}^4 = \{((1,1,1),1),\ ((1,0,0),1),\ ((1,1,0),0),\ ((0,0,1),0)\}.$$

(i) Suppose we run the ID3 algorithm up to depth 2 (namely, we pick the root node and its children according to the algorithm, but instead of keeping on with the recursion, we stop and pick leaves according to the majority label in each subtree). Assume that the information gain (equivalently, conditional entropy) is used to measure the quality of each feature and if two features get the same score, one of them is picked arbitrarily. Show that the training error of the resulting decision tree is at least 1/4.

(ii) Find a decision tree of depth 2 that attains zero training error.

Solution

(i) Let $E(p) = -p\log p - (1-p)\log(1-p)$ be the entropy of $\mathrm{Ber}(p)$ (Bernoulli distribution with probability p). The ID3 algorithm proceeds as follows.

(1. root node) Calculate entropy of y conditional on each feature:

$$H(y|x_1) = \frac{1}{4}\cdot 0 + \frac{3}{4}\cdot E(2/3) \approx 0.478,$$
$$H(y|x_2) = \frac{1}{2}E(1/2) + \frac{1}{2}E(1/2) \approx 0.693,$$
$$H(y|x_3) = \frac{1}{2}E(1/2) + \frac{1}{2}E(1/2) \approx 0.693.$$

Therefore, feature x_1 minimizes the entropy (equivalently, maximizes the information gain).

(1.1 internal node after $x_1 = 0$). Only one training example is left. This example has $y = 0$. Therefore, we designate the current node as a leaf and assign to it the prediction $y = 0$.

(1.2 internal node after $x_1 = 1$). Calculate entropy of y conditional on each remaining feature and the event $x_1 = 1$:

$$H(y|x_2, x_1 = 1) = \frac{1}{3}\cdot 0 + \frac{2}{3}\cdot E(1/2) \approx 0.520,$$
$$H(y|x_3, x_1 = 1) = \frac{1}{3}\cdot 0 + \frac{2}{3}\cdot E(1/2) \approx 0.520.$$

Therefore, either of the two features x_2, x_3 minimizes the entropy. Suppose we pick x_2.

(1.2.1 leafs). Both after $x_2 = 0$ and after $x_2 = 1$, we are left with two examples having different labels. Therefore, we assign prediction labels arbitrarily, say, $y = 1$ to $x_2 = 0$ and $y = 0$ to $x_2 = 1$. Algorithm stops.

The resulting decision tree misclassifies example 1, but correctly classifies the three remaining examples. Therefore, its training error is $\frac{1}{4}$. Besides, it is easy to show that, if different arbitrary decisions were made during fitting, the resulting trees would still have the traning error equal to $\frac{1}{4}$.

(ii) The following decision tree of depth 2 has zero training error.

if $x_2 = 1$: if $x_3 = 1$:
predict $y = 1$ else: predict $y = 0$ else: if $x_3 = 1$: predict $y = 0$ else: predict $y = 1$

Problem 2 (survival on the *Titanic*)

The titanic.xls dataset contains information on each of the 1309 passengers of RMS *Titanic*. The goal is to predict passenger survival. Use the first 1100 rows as the training sample and the remaining rows as the test sample.

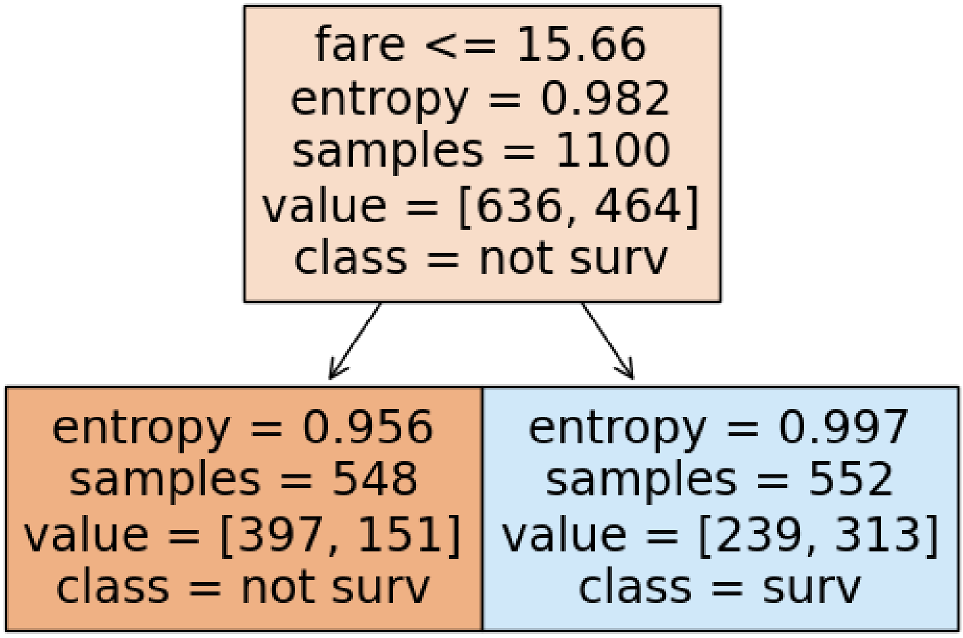
(i) Fit a decision tree of maximal depth $d \in \{1, 2, 3, 4, 5, 6, 7, 8\}$ with the information gain as the splitting criterion.

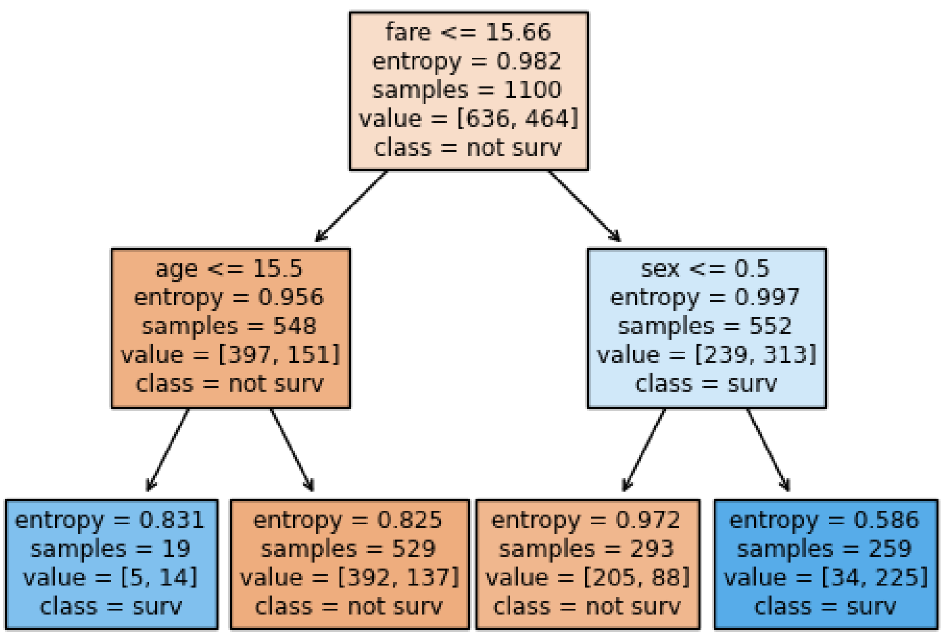
(ii) Plot the decision trees corresponding to $d = 1$ and $d = 2$ (in Python, use sklearn.tree.plot_tree). Interpret the results.

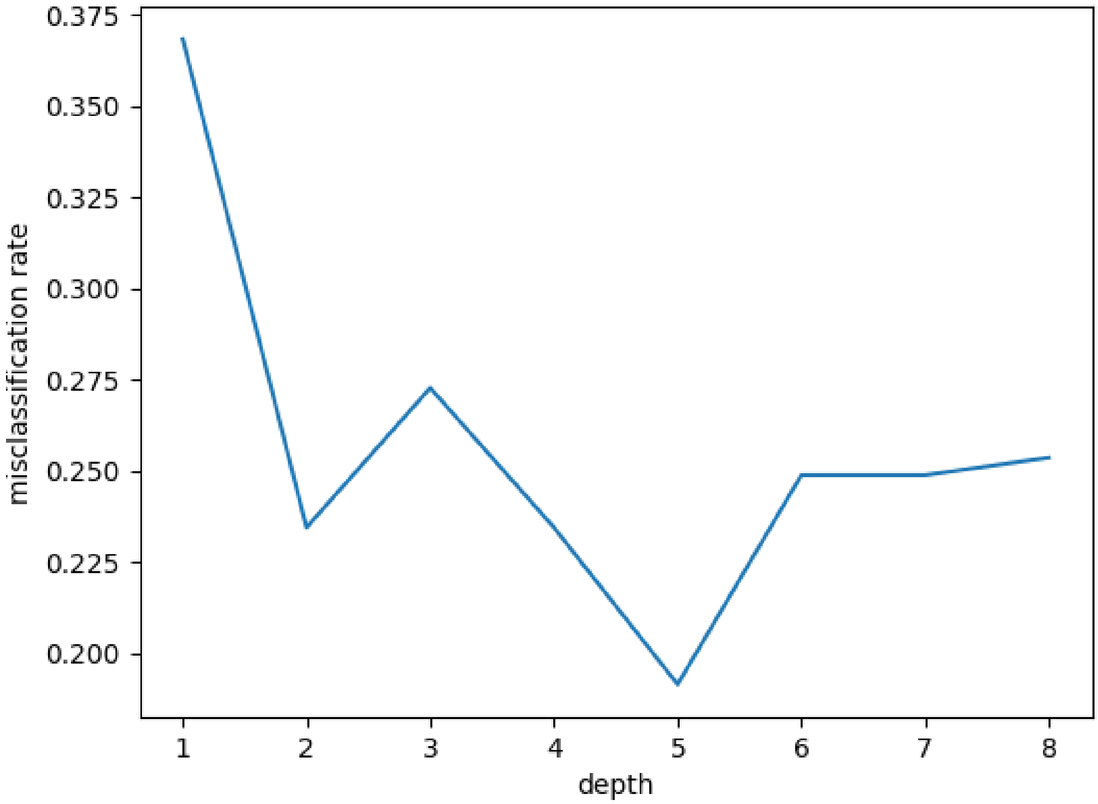
(iii) Calculate the test error (misclassification rate) on the test sample and plot it as a function of depth d. Does the test error change much? Which value of d would you choose?

[4]:
```
# SOLUTION FOR PROBLEM 2

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import zero_one_loss

# import data
datapath = '/Users/franguri/Library/CloudStorage/Dropbox/_TEACHING_/Econ 425 ML␣
↪UCLA Winter 2024/WEEK 4 Decision trees/titanic.xls'
data = pd.read_excel(datapath)

X = data[['pclass','sex','age','sibsp','parch','fare']].copy()
X['sex_dummy'] = (X['sex'] == 'female').astype(int)
X = X.drop(columns=['sex'])
X.columns = ['pclass','age','siblings','parch','fare','sex']
y = data['survived']

X_train = X[:1100]
y_train = y[:1100]
X_test = X[1100:]
y_test = y[1100:]
```

```
d_range = [1,2,3,4,5,6,7,8]
misclass_rates = np.empty((len(d_range),))
for i in range(len(d_range)):
    # fit a decision tree
    clf = DecisionTreeClassifier(criterion='entropy', max_depth=d_range[i],␣
 ↪random_state=0)
    clf.fit(X_train, y_train)

    # plot the decision tree
    if d_range[i] in [1,2]:
        plt.figure()
        plot_tree(clf, feature_names=X.columns, class_names=['not surv',␣
 ↪'surv'], filled=True)
        plt.show()

    # calculate test error:
    y_pred = clf.predict(X_test)
    misclass_rates[i] = zero_one_loss(y_test, y_pred)

# plot misclassification rates:
plt.figure()
plt.plot(d_range, misclass_rates)
plt.xlabel('depth')
plt.ylabel('misclassification rate')
plt.show()
```


fare <= 15.66
entropy = 0.982
samples = 1100
value = [636, 464]
class = not surv
entropy = 0.956
samples = 548
value = [397, 151]
class = not surv
entropy = 0.997
samples = 552
value = [239, 313]
class = surv


fare <= 15.66
entropy = 0.982
samples = 1100
value = [636, 464]
class = not surv
age <= 15.5
entropy = 0.956
samples = 548
value = [397, 151]
class = not surv
sex <= 0.5
entropy = 0.997
samples = 552
value = [239, 313]
class = surv
entropy = 0.831
samples = 19
value = [5, 14]
class = surv
entropy = 0.825
samples = 529
value = [392, 137]
class = not surv
entropy = 0.972
samples = 293
value = [205, 88]
class = not surv
entropy = 0.586
samples = 259
value = [34, 225]
class = surv



0.375
0.350
0.325
0.300
0.275
0.250
0.225
0.200
misclassification rate
1
2
3
4
5
6
7
8
depth