ECON 425 HW4

January 30, 2024

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?

ML HW 4

February 7, 2024

0.1 Problem 2

[118]:
```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, f1_score
import numpy as np
from sklearn.metrics import confusion_matrix, roc_curve, roc_auc_score, auc
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import numpy as np
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, roc_auc_score
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import accuracy_score
```

[119]:
```
df=pd.read_csv("/Users/sandinatatu/Desktop/titanic.csv")
df=df[:-1]
```

0.1.1 i)

Prior to fitting decision trees to my dataset, I transform the dataset by obtaining dummies for all of the non-numerical variables.

[124]:
```
X = df.drop(['survived','name'], axis=1)
y=df[["survived"]].copy()

cols = X.columns[X.dtypes=="object"]

for col in cols:
    X[col] = X[col].fillna('missing')

for col in cols:
    X[col] = pd.Categorical(X[col])

X=pd.get_dummies(X, columns= cols, prefix=cols, drop_first = True, dtype=int)
```

[125]:
```
x_train = X.iloc[:1100]
x_test = X.iloc[1100:]

y_train = y.iloc[:1100]
y_test = y.iloc[1100:]
```

Below I fit decision treee of maximal depth $\{1, 2, 3, 4, 5, 6, 7, 8\}$ to my dataset. I use the information gain (or entropy) as the splitting criterion. The best out-of-sample accuracy is 0.98, obtained for d $\{2, 3, 4, 8\}$. The best in-sample accuracy is 0.98, obtained for d=8.

[128]:
```
stored_models={}
misclassification_rates={}

np.random.seed(15)

for d in range(1,9):
    clf =tree.DecisionTreeClassifier(max_depth = d, criterion = "entropy")
    clf.fit(x_train, y_train)
    y_pred = clf.predict(x_train)
    acc=accuracy_score(y_train, y_pred)

    print(f"In sample accuracy for max_depth = {d} is {round(acc,2)}")

    y_pred = clf.predict(x_test)
    acc=accuracy_score(y_test,y_pred)
    misclassification_rates[d] = 1-acc

    print(f"Out of Sample accuracy for max_depth = {d} is {round(acc,2)}")

    if d in[1,2]:
        stored_models[d] = clf
```

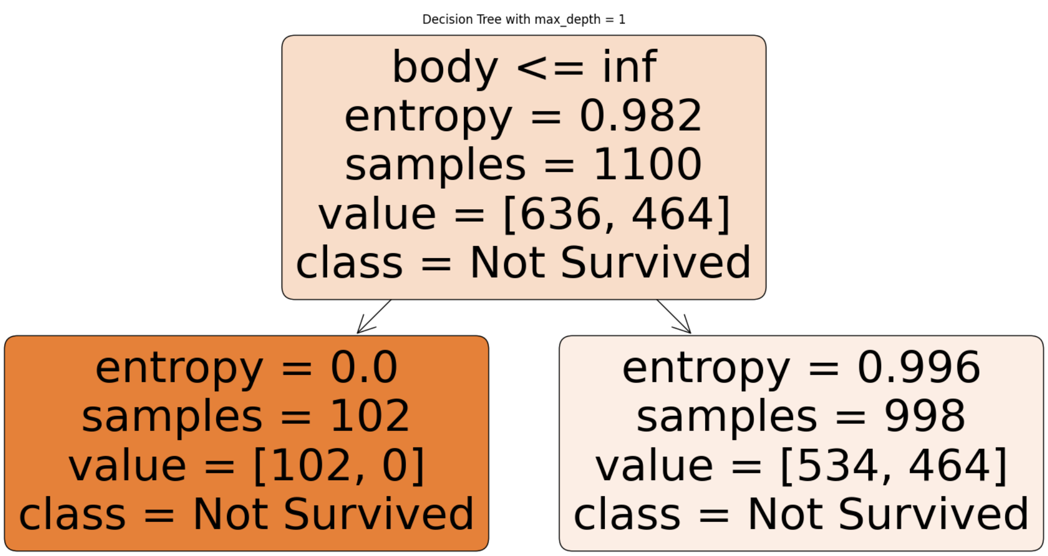
```
In sample accuracy for max_depth = 1 is 0.58
Out of Sample accuracy for max_depth = 1 is 0.83
In sample accuracy for max_depth = 2 is 0.97
Out of Sample accuracy for max_depth = 2 is 0.98
In sample accuracy for max_depth = 3 is 0.97
Out of Sample accuracy for max_depth = 3 is 0.98
In sample accuracy for max_depth = 4 is 0.97
Out of Sample accuracy for max_depth = 4 is 0.98
In sample accuracy for max_depth = 5 is 0.91
Out of Sample accuracy for max_depth = 5 is 0.84
In sample accuracy for max_depth = 6 is 0.78
Out of Sample accuracy for max_depth = 6 is 0.78
In sample accuracy for max_depth = 7 is 0.9
Out of Sample accuracy for max_depth = 7 is 0.79
In sample accuracy for max_depth = 8 is 0.98
Out of Sample accuracy for max_depth = 8 is 0.98
```

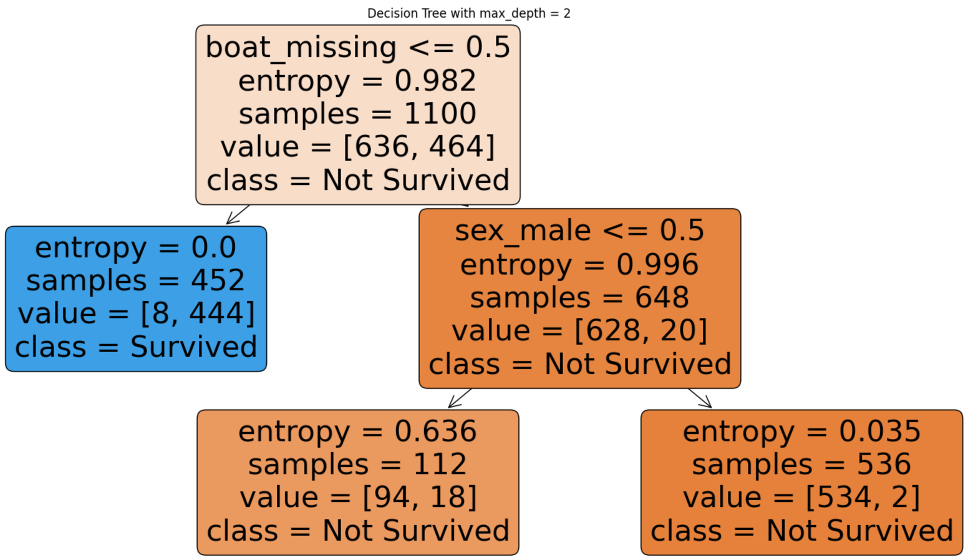
0.1.2 ii)

Below I plot the decision trees with max_depths 1 and 2. For max_depth 1, the tree will always predict that someone did not survive, hence the low in-sample accuracy. For max_depth of 2, the accuracy is much higher. This is because this tree initially uses the boat_missing criterion, which allows it to identify passengers who had boats and categorize them as survivors. It then does another split for those who had boat boat_missing = 1, and splits the further based on their sex.

[130]:
```
import matplotlib.pyplot as plt
from sklearn import tree

for d in [1, 2]:
    clf = stored_models[d]
    plt.figure(figsize=(20,10))
    tree.plot_tree(clf, filled=True, rounded=True,
                   class_names=["Not Survived", "Survived"],
                   feature_names=list(x_train.columns))
    plt.title(f"Decision Tree with max_depth = {d}")
    plt.show()
```



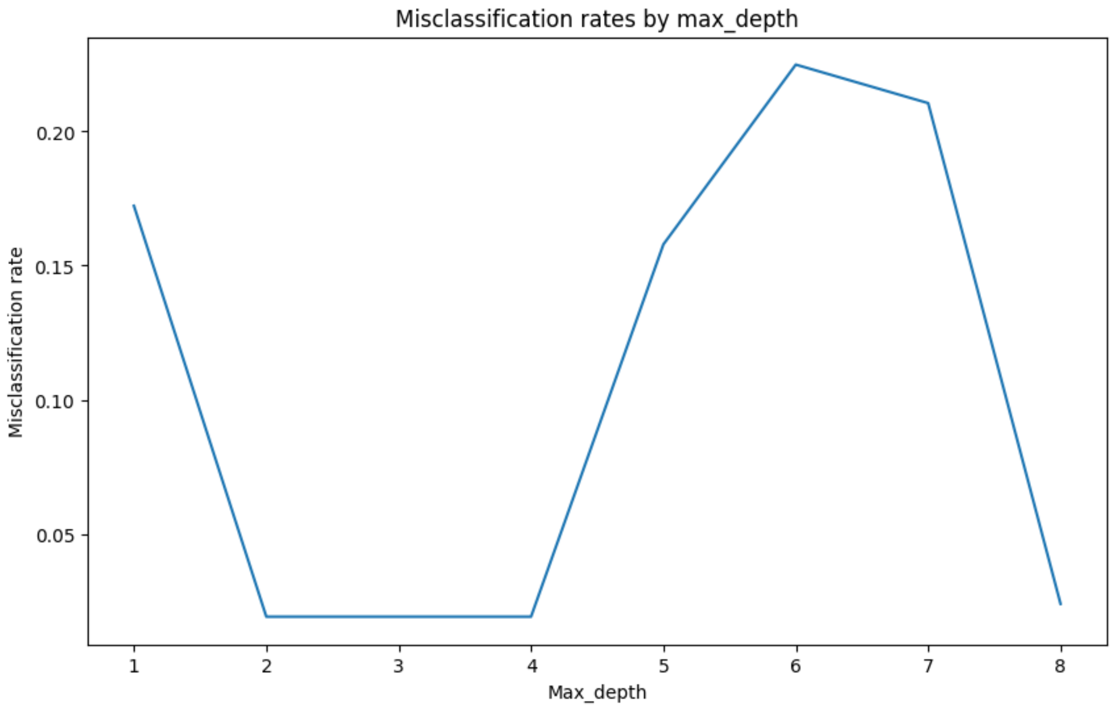


0.1.3 iii)

[145]:
```python
keys = list(misclassification_rates.keys())
values = list(misclassification_rates.values())
```

Below I plot the misclassification rates for each max_depth. The lowes misclassification rates are obtained for d {2, 3, 4, 8}. The test error does indeed change a lot, with it being much higher for d=1 or when d {5,6,7}. Overall, I would choose d=1 for the purpose of parsimony.

[148]:
```python
plt.figure(figsize=(10,6))
plt.plot(keys, values)
plt.title("Misclassification rates by max_depth")
plt.xlabel("Max_depth")
plt.ylabel("Misclassification rate")
plt.show()
```


Misclassification rates by max_depth
Misclassification rate
0.20
0.15
0.10
0.05
1
2
3
4
5
6
7
8
Max_depth