Applied Machine Learning

Assignment 4

Names :

Sema Abdelnasser Mosaad

Nada Mohammed Zakaria

Dina Ibrahim Mohammady

Part 1

a)

- **For weather feature:**

  Gini (Cloudy) = $1 - [(\frac{1}{3})^2 + (\frac{2}{3})^2] = \frac{4}{9}$

  Gini (Sunny) = $1 - [(\frac{4}{4})^2 + (\frac{0}{4})^2] = 0$

  Gini (Rainy) = $1 - [(\frac{2}{3})^2 + (\frac{1}{3})^2] = \frac{4}{9}$

  Gini (Children) = $(\frac{3}{10} * \frac{4}{9}) + (\frac{4}{10} * 0) + (\frac{3}{10} * \frac{4}{9})$ = 0.266667

- **For temperature feature:**

  Gini (Hot) = $1 - [(\frac{3}{4})^2 + (\frac{1}{4})^2] = \frac{3}{8}$

  Gini (Mild) = $1 - [(\frac{2}{4})^2 + (\frac{2}{4})^2] = \frac{1}{2}$

  Gini (Cool) = $1 - [(\frac{1}{1})^2 + (\frac{0}{1})^2] = 0$

  Gini (Cold) = $1 - [(\frac{1}{1})^2 + (\frac{0}{1})^2] = 0$

  Gini (Children) = $(\frac{4}{10} * \frac{3}{8}) + (\frac{4}{10} * \frac{1}{2})$ = 0.35

- **For Humidity feature:**

  Gini (High) = $1 - [(\frac{5}{6})^2 + (\frac{1}{6})^2] = \frac{5}{18}$

  Gini (Normal) = $1 - [(\frac{3}{4})^2 + (\frac{1}{4})^2] = \frac{3}{8}$

  Gini (Children) = $(\frac{6}{10} * \frac{5}{18}) + (\frac{4}{10} * \frac{3}{8})$ = 0.316667

➢ **For Wind feature:**

Gini (Strong) = 1 - $[(\frac{2}{7})^2 + (\frac{5}{7})^2] = \frac{20}{49}$

Gini (Weak) = 1 - $[(\frac{2}{3})^2 + (\frac{1}{3})^2] = \frac{4}{9}$

Gini (Children) = $(\frac{7}{10} * \frac{20}{49}) + (\frac{3}{10} * \frac{4}{9})$ = 0.41905

After calculating Gini for all features, we will find that Gini (Weather) is the least so Weather feature is the root of the tree.

The new dataset:

Weather	Temperature	Humidity	Wind	Hiking
Cloudy	Hot	High	Strong	No
Rainy	Cold	Normal	Strong	Yes
Cloudy	Mild	Normal	Strong	Yes
Rainy	Cool	Normal	Strong	No
Cloudy	Mild	High	Weak	Yes
Rainy	Hot	Normal	Weak	Yes

- **For Temperature feature:**

Gini (Hot) = 1 - $[(\frac{1}{2})^2 + (\frac{1}{2})^2] = \frac{1}{2}$

Gini (Mild) = 1 - $[(\frac{2}{2})^2 + (\frac{0}{2})^2] = 0$

Gini (Cool) = 1 - $[(\frac{1}{1})^2 + (\frac{0}{1})^2] = 0$

Gini (Cold) = 1 - $[(\frac{1}{1})^2 + (\frac{0}{1})^2] = 0$

Gini (Children) = $(\frac{2}{6} * \frac{1}{2})$ = 0.166667

- **For Humidity feature:**

Gini (High) = 1 - $[(\frac{1}{2})^2 + (\frac{1}{2})^2] = \frac{1}{2}$

Gini (Normal) = 1 - $[(\frac{3}{4})^2 + (\frac{1}{4})^2] = \frac{3}{8}$

Gini (Children) = $(\frac{2}{6} * \frac{1}{2}) + (\frac{4}{6} * \frac{3}{8})$ = 0.416667

- **For wind feature:**

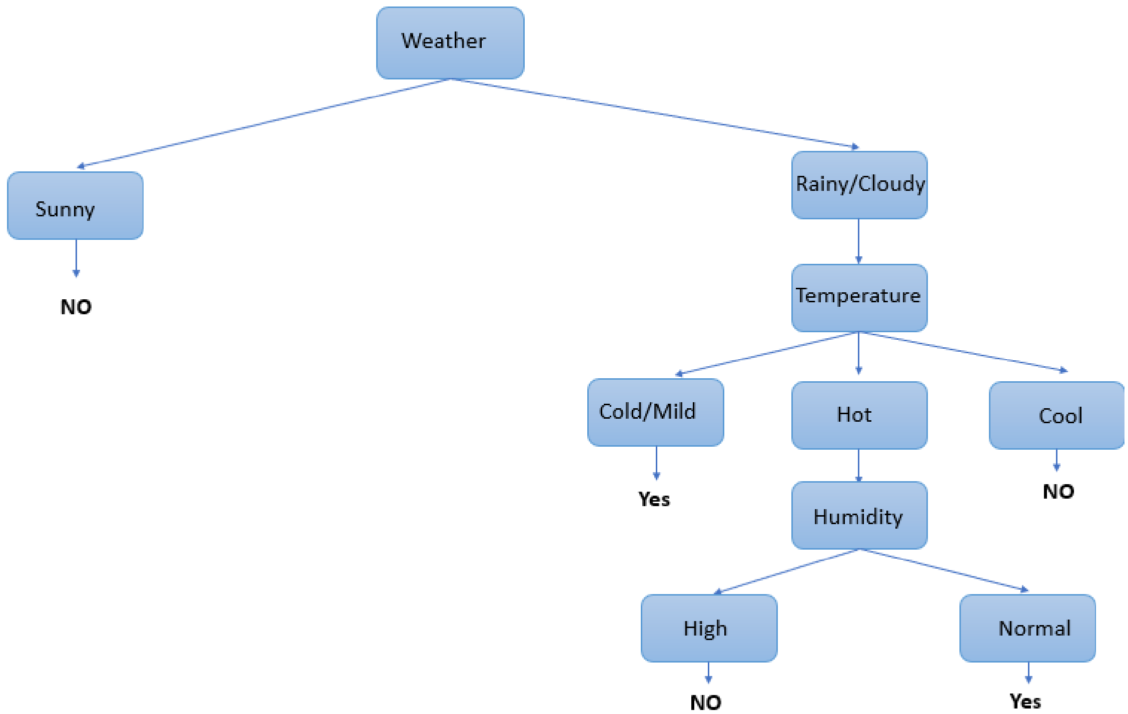
Gini (Strong) = 1 - $[(\frac{2}{4})^2 + (\frac{2}{4})^2] = \frac{1}{2}$

Gini (Normal) = 1 - $[(\frac{2}{2})^2 + (\frac{2}{0})^2] = \frac{3}{8}$

Gini (Children) = $(\frac{4}{6} * \frac{1}{2})$ = 0.333333

After the second iteration the Temperature feature has the lowest Gini, so it is the intermediate branch.

And other features have known labels.



b)

E(S) = $\frac{-6}{10}\log\frac{6}{10} - \frac{4}{10}\log\frac{4}{10}$ = 0.97095

G (S, Weather) = 0.97095 - $\frac{3}{10}(\frac{-1}{3}\log\frac{1}{3} - \frac{2}{3}\log\frac{2}{3})$ – 0

- $\frac{3}{10}(\frac{-2}{3}\log\frac{2}{3} - \frac{1}{3}\log\frac{1}{3})$ =

0.97095 – 0.27548 – 2.7548 = 0.41999

G (S, Temperature) = 0.97095 - $\frac{4}{10}(\frac{-3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4})$

- $\frac{4}{10}(\frac{-2}{4}\log\frac{2}{4} - \frac{2}{4}\log\frac{2}{4})$ - $\frac{1}{10}$ (- log) - $\frac{1}{10}$(- log) =

0.97095 – 0.3245 –0.4= 0.24645

G (S, Humidity) = 0.97095 - $\frac{6}{10}(\frac{-5}{6}\log\frac{5}{6} - \frac{1}{6}\log\frac{1}{6})$ - $\frac{4}{10}(\frac{-3}{4}\log\frac{3}{4} - \frac{1}{4}\log\frac{1}{4})$

= 0.97095 – 0.390013 –0.32451= 0.256427

G (S, Wind) = 0.97095 - $\frac{7}{10}(\frac{-5}{7}\log\frac{5}{7} - \frac{2}{7}\log\frac{2}{7})$ - $\frac{3}{10}(\frac{-1}{3}\log\frac{1}{3} - \frac{2}{3}\log\frac{2}{3})$

= 0.97095 – 0.60418 –0.27548= 0.09129

$$E(S) = \frac{-2}{6} \log \frac{2}{6} - \frac{-4}{6} \log \frac{4}{6} = 0.91829$$

$$G\ (S, \text{Temperature}) = 0.91829 - \frac{2}{6}\left(\frac{-1}{2} \log \frac{1}{2} - \frac{-1}{2} \log \frac{1}{2} \right) = 0.5849$$

$$G\ (S, \text{Humidity}) = 0.91829 - \frac{2}{6}\left(\frac{-1}{2} \log \frac{1}{2} - \frac{-1}{2} \log \frac{1}{2} \right)$$

$$- \frac{4}{6}\left(\frac{-3}{4} \log \frac{3}{4} - \frac{-1}{4} \log \frac{1}{4} \right) = 0.91829 - 0.33333 - 0.54085 = 0.0441$$

c)

Both the Gini Index and Information Gain are commonly used metrics in decision tree algorithms for evaluating the quality of a split. While both metrics serve the same purpose, they have different characteristics**.**

Gini Index:

Advantages:

1. Simplicity: The Gini Index is a straightforward metric that measures the impurity or inequality in a given set of data. It is easy to understand and compute.

2. Computationally efficient: Calculating the Gini Index involves simple arithmetic operations, making it computationally efficient compared to other metrics.

3. Robust to imbalanced datasets: The Gini Index tends to work well with imbalanced class distributions, as it focuses on misclassifications across all classes.

Disadvantages:

1. Ignores information gain: The Gini Index only considers the impurity of classes in a dataset and does not explicitly consider the information gain obtained by splitting the data based on a particular feature.

2. Biased towards multi-class classification: The Gini Index has a tendency to favor features with a large number of distinct classes. This can be a disadvantage in binary classification tasks or when dealing with features with a small number of classes.

Information Gain:

Advantages:

1. Incorporates information gain: Information Gain measures the reduction in entropy or uncertainty after splitting the data based on a feature. It explicitly considers the information gained by the split, which can be beneficial for feature selection.

2. Suitable for binary classification: Information Gain is particularly well-suited for binary classification tasks, where it aims to maximize the purity or homogeneity of each class.

3. Handles features with different numbers of classes: Information Gain is not biased towards features with a large number of classes, making it suitable for both binary and multi-class classification problems.

Disadvantages:

1. Sensitive to the number of classes: Information Gain tends to favor features with a large number of classes, which can result in biased feature selection in multi-class classification tasks.

2. Prone to overfitting: Information Gain may lead to overfitting, especially when dealing with noisy or irrelevant features that create spurious information gain.

Step 1: Data Preparation and Exploration

The initial step involves loading the dataset and exploring its structure

```
[ ]  import pandas as pd
     import numpy as np
     from sklearn.preprocessing import MinMaxScaler
     from sklearn.feature_selection import SelectKBest, f_classif
     from sklearn.model_selection import train_test_split
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.metrics import accuracy_score
     from sklearn.metrics import classification_report
     from sklearn.metrics import confusion_matrix
     import matplotlib.pyplot as plt
```

```
data.info()
```

```
data.head()
```

	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	...	dst_host_srv_count	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_
0	0	181	5450	0	0	0	0	0	1	0	...	9	1.0	0.0	0.11	
1	0	239	486	0	0	0	0	0	1	0	...	19	1.0	0.0	0.05	
2	0	235	1337	0	0	0	0	0	1	0	...	29	1.0	0.0	0.03	
3	0	219	1337	0	0	0	0	0	1	0	...	39	1.0	0.0	0.03	
4	0	217	2032	0	0	0	0	0	1	0	...	49	1.0	0.0	0.02	

5 rows × 39 columns

Step 2: Feature Selection

The features (X) and the target variable (Y) are separated from the dataset.

The features are normalized using the MinMaxScaler to ensure all features have a similar scale.

```
[ ]  #  X and Y
     X = data.iloc[:, :-1] # all columns except the last one
     Y = data.iloc[:, -1] # last column
```

```
[ ]  # Normalize X using MinMaxScaler
     scaler = MinMaxScaler()
     X = scaler.fit_transform(X)
```

```
Y
```

```
0         0
1         0
2         0
3         0
4         0
         ..
494016    0
494017    0
494018    0
494019    0
494020    0
Name: target, Length: 494021, dtype: int64
```

```
X
```

```
array([[0.00000000e+00, 2.61041764e-07, 1.05713002e-03, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 3.44690506e-07, 9.42688423e-05, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 3.38921627e-07, 2.59336301e-04, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       ...,
       [0.00000000e+00, 2.92770597e-07, 2.32762574e-04, ...,
        1.00000000e-02, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 4.19685930e-07, 2.32762574e-04, ...,
        1.00000000e-02, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 3.15846112e-07, 2.39357513e-04, ...,
        1.00000000e-02, 0.00000000e+00, 0.00000000e+00]])
```

Step 3: Subset Creation

- Three subsets are created from the selected features and target variable using train_test_split with different test sizes.

- Subset 1: 70% train, 30% test

- Subset 2: 60% train, 40% test

- Subset 3: 50% train, 50% test

```
Subset 1:
Number of training examples: 345814
Number of test examples: 148207
Training labels distribution: (array([0, 1]), array([ 68086, 277728]))
Test labels distribution: (array([0, 1]), array([ 29192, 119015]))
```

```
Subset 2:
Number of training examples: 296412
Number of test examples: 197609
Training labels distribution: (array([0, 1]), array([ 58301, 238111]))
Test labels distribution: (array([0, 1]), array([ 38977, 158632]))
```

```
Subset 3:
Number of training examples: 247010
Number of test examples: 247011
Training labels distribution: (array([0, 1]), array([ 48628, 198382]))
Test labels distribution: (array([0, 1]), array([ 48650, 198361]))
```

Step 4: Classification without Mitigation Strategies

- Decision Tree Classifier is trained on each subset without any mitigation strategies.

- The classifier's performance is evaluated on the test data using accuracy, precision, recall, and F1-score.

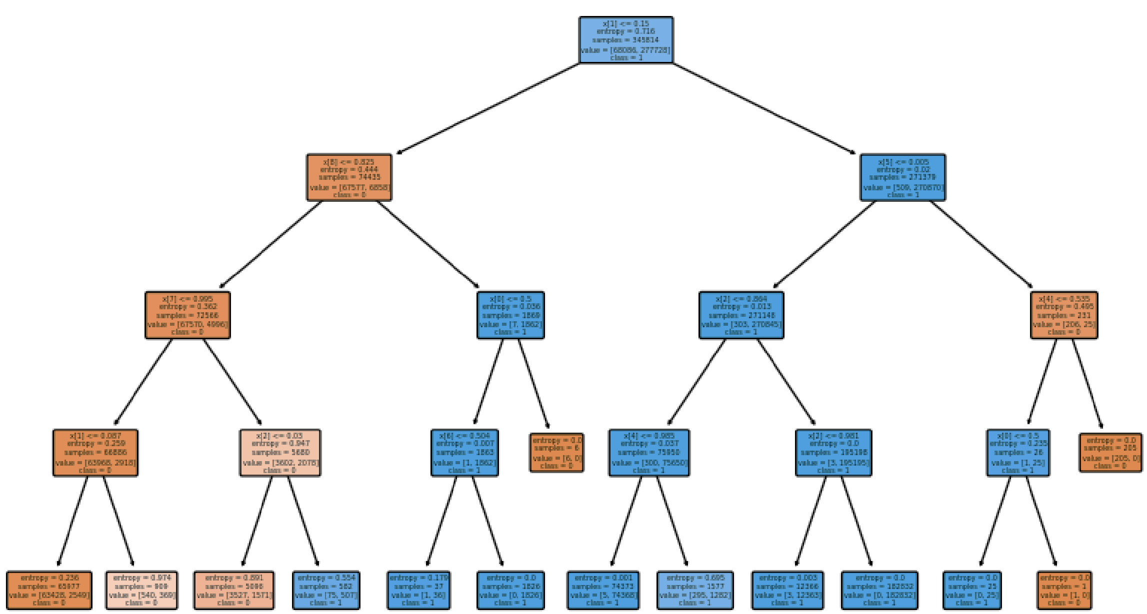
```
Subset 1:
Accuracy: 0.9905065212844197
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     29192
           1       1.00      0.99      0.99    119015

    accuracy                           0.99    148207
   macro avg       0.98      0.99      0.99    148207
weighted avg       0.99      0.99      0.99    148207
```

```
Subset 2:
Accuracy: 0.9908050746676518
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     38977
           1       1.00      0.99      0.99    158632

    accuracy                           0.99    197609
   macro avg       0.98      0.99      0.99    197609
weighted avg       0.99      0.99      0.99    197609
```

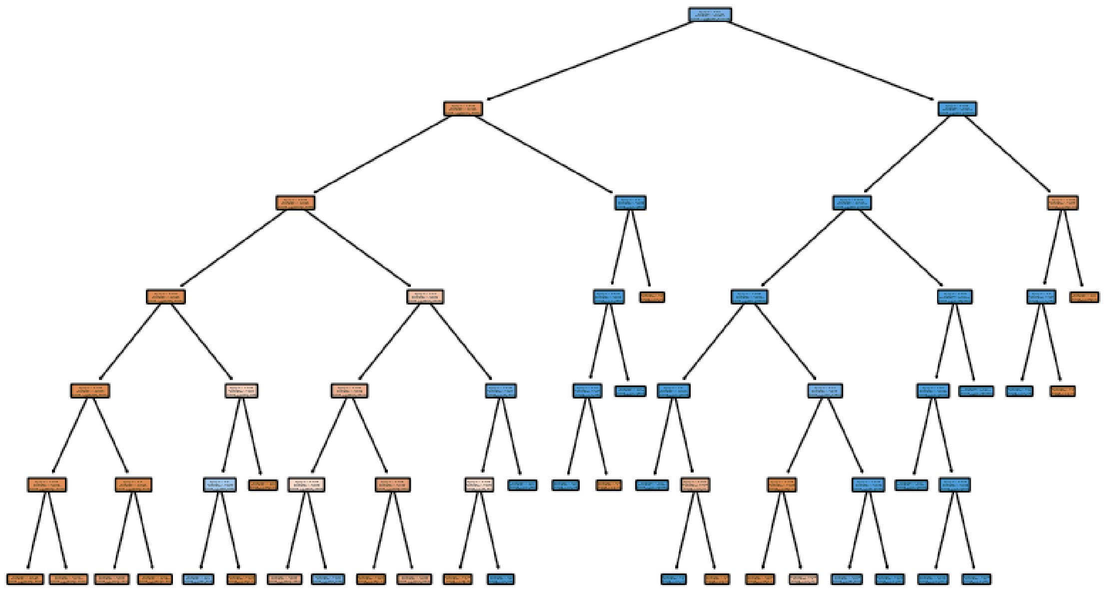
```
Subset 3:
Accuracy: 0.9909072875297051
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     48650
           1       1.00      0.99      0.99    198361

    accuracy                           0.99    247011
   macro avg       0.98      0.99      0.99    247011
weighted avg       0.99      0.99      0.99    247011
```

-Visualize the best split of the Decision tree by considering Entropy as a measure of node impurity and assuming parameters max depth=[4, 6, 8]

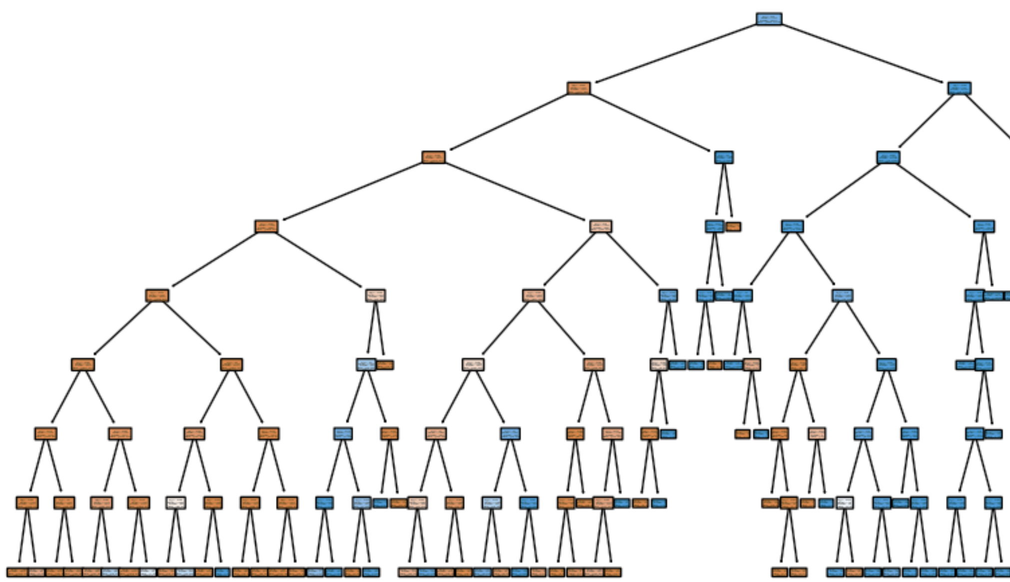
Subset 1, Max depth 4



Subset 1, Max depth 6



Subset 1, Max depth 8



Subset 1:

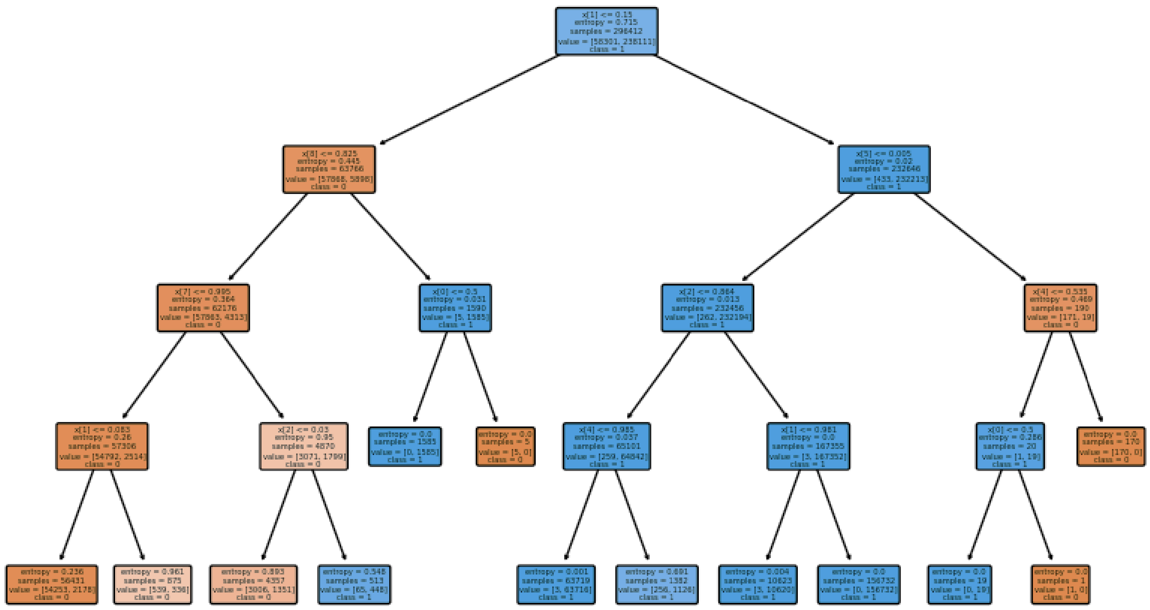
Max depth 4: accuracy = 0.9857226716686797

Max depth 6: accuracy = 0.988522809314

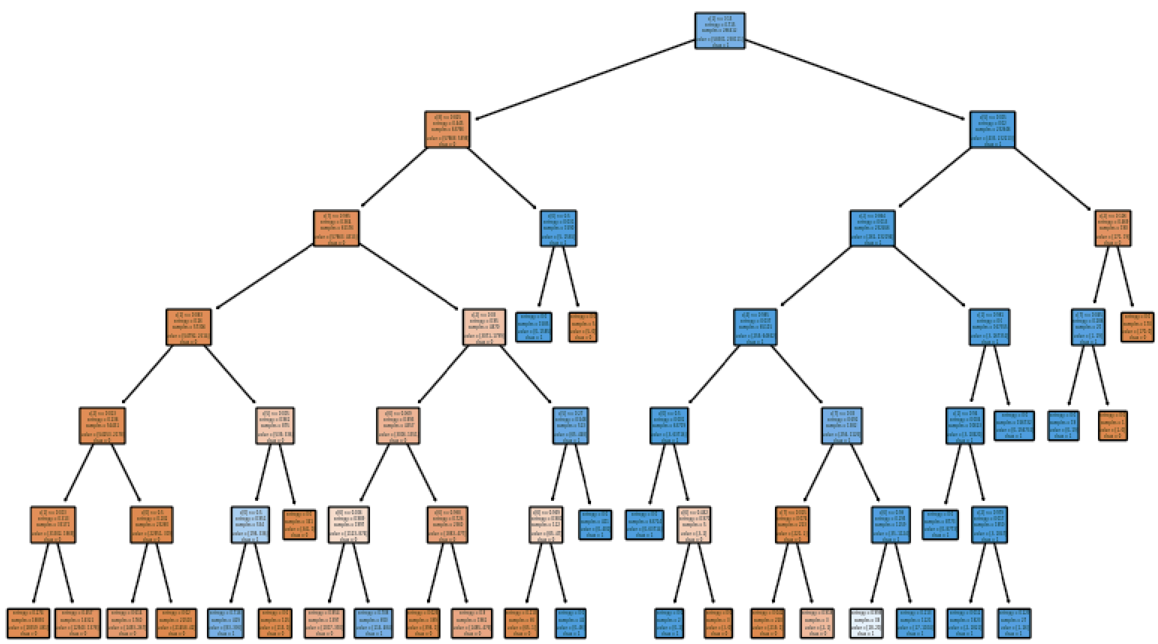
Max depth 8: accuracy = 0.9901219240656649

Best split for Subset 1 (Entropy): max depth = 8, accuracy = 0.9901219240656649

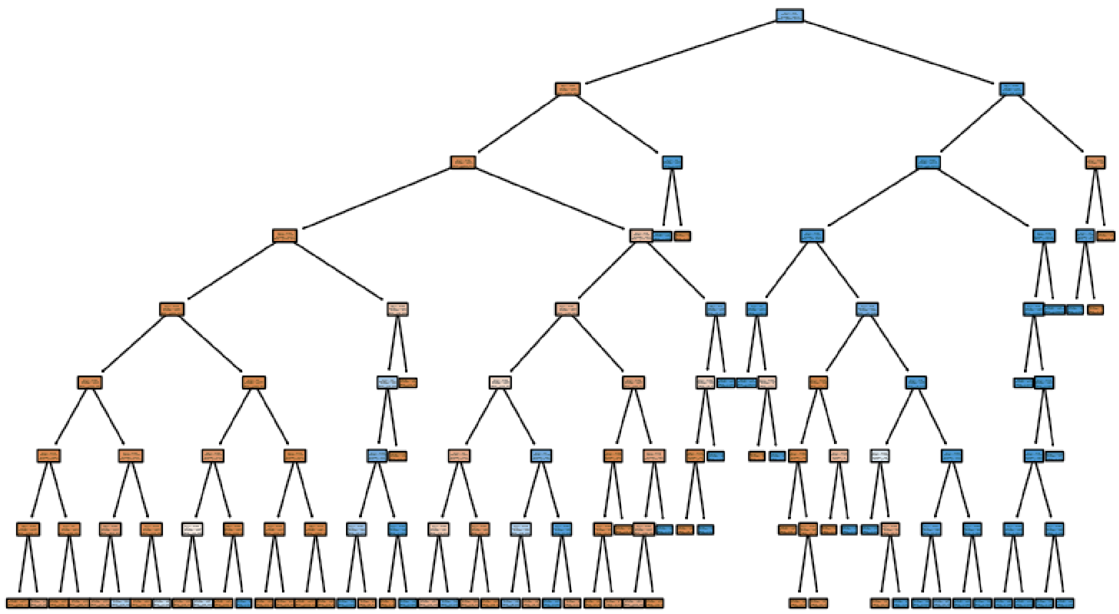
Subset 2, Max depth 4



Subset 2, Max depth 6



Subset 2, Max depth 8



Subset 2:

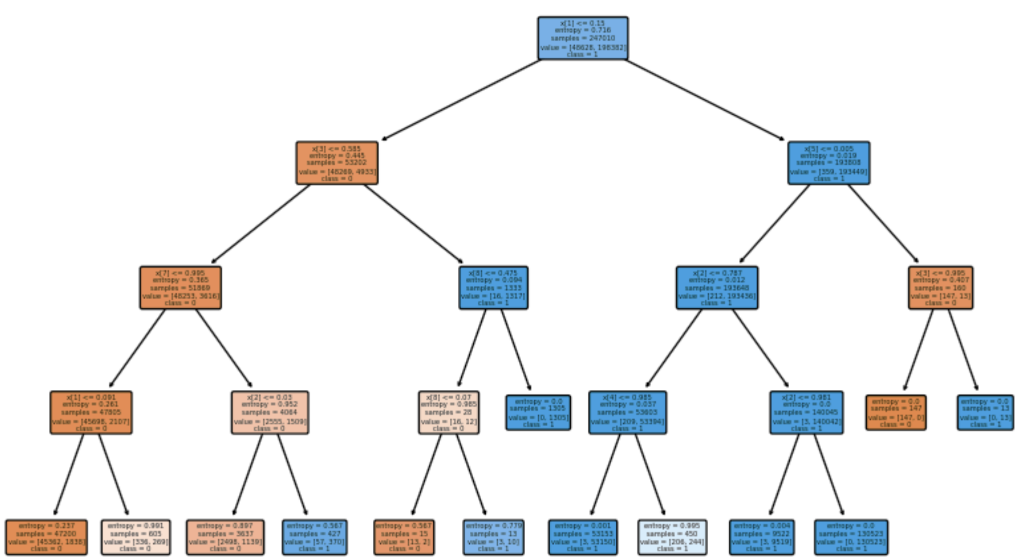
Max depth 4: accuracy = 0.98587108886740999

Max depth 6: accuracy = 0.9887758148667318

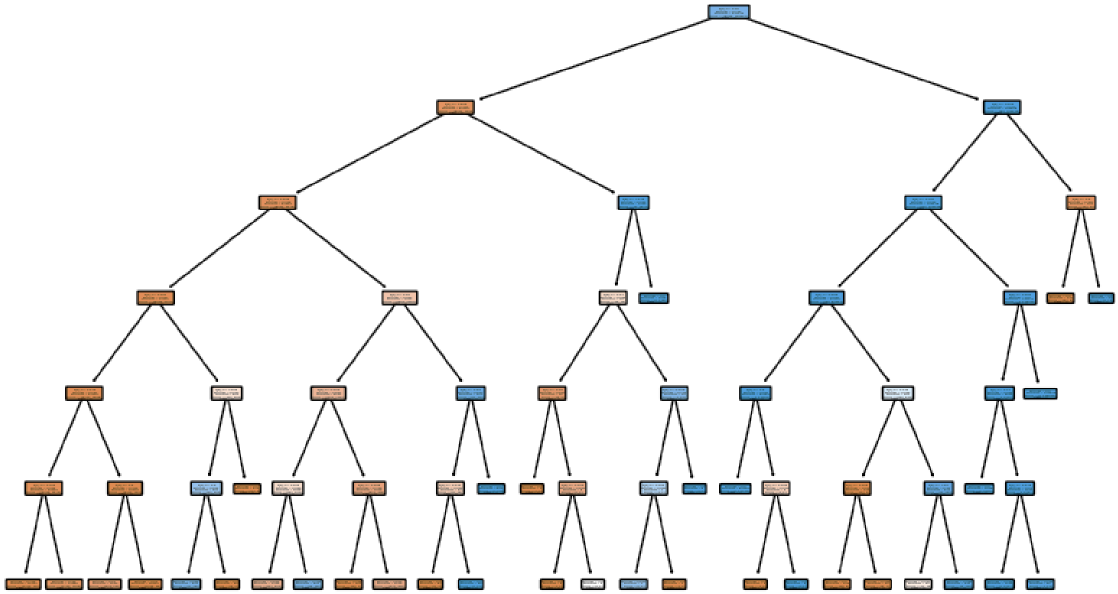
Max depth 8: accuracy = 0.9902484198594194

Best split for Subset 2 (Entropy): max depth = 8, accuracy = 0.9902484198594194

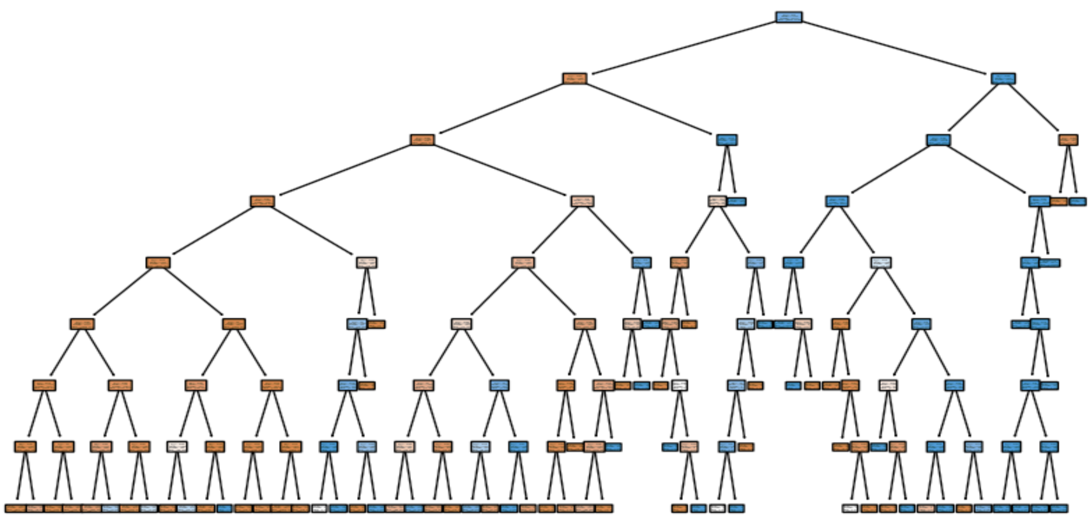
Subset 3, Max depth 4



Subset 3, Max depth 6



Subset 3, Max depth 8



Subset 3:

Max depth 4: accuracy = 0.986073494702665

Max depth 6: accuracy = 0.9889073765945646

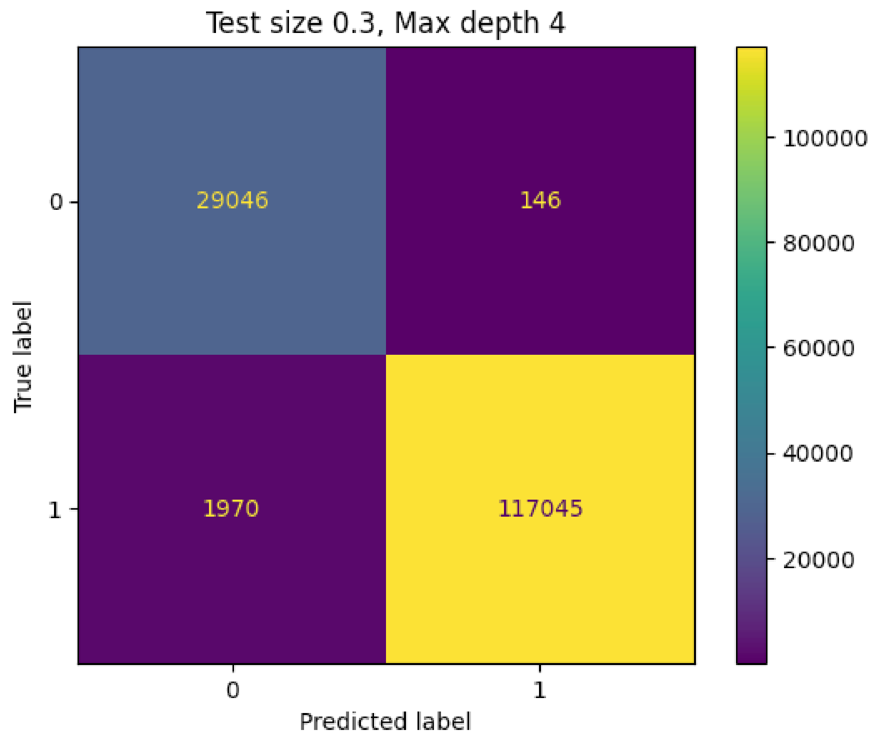
Max depth 8: accuracy = 0.9904093339972714

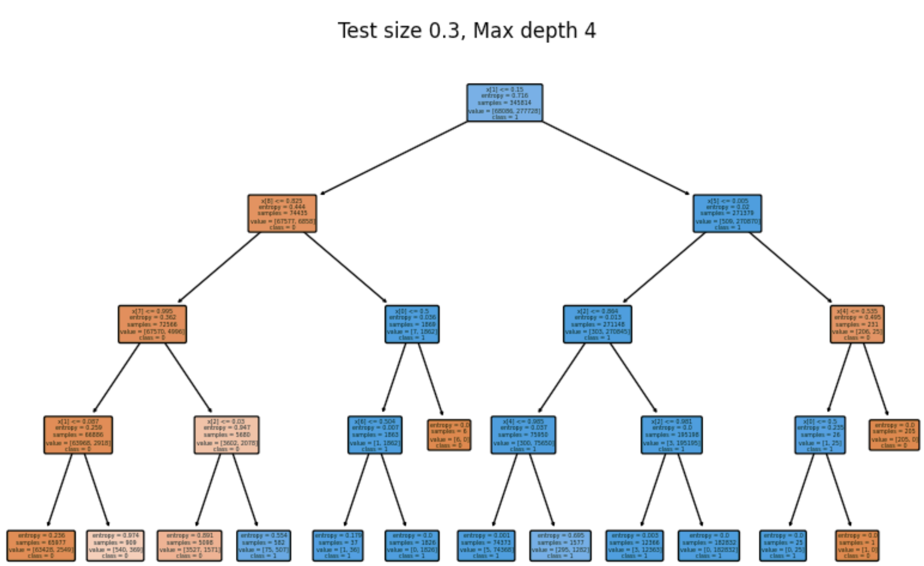
Best split for Subset 3 (Entropy): max depth = 8, accuracy = 0.9904093339972714

Best split from all subsets (Entropy): Subset 1, max depth = 8

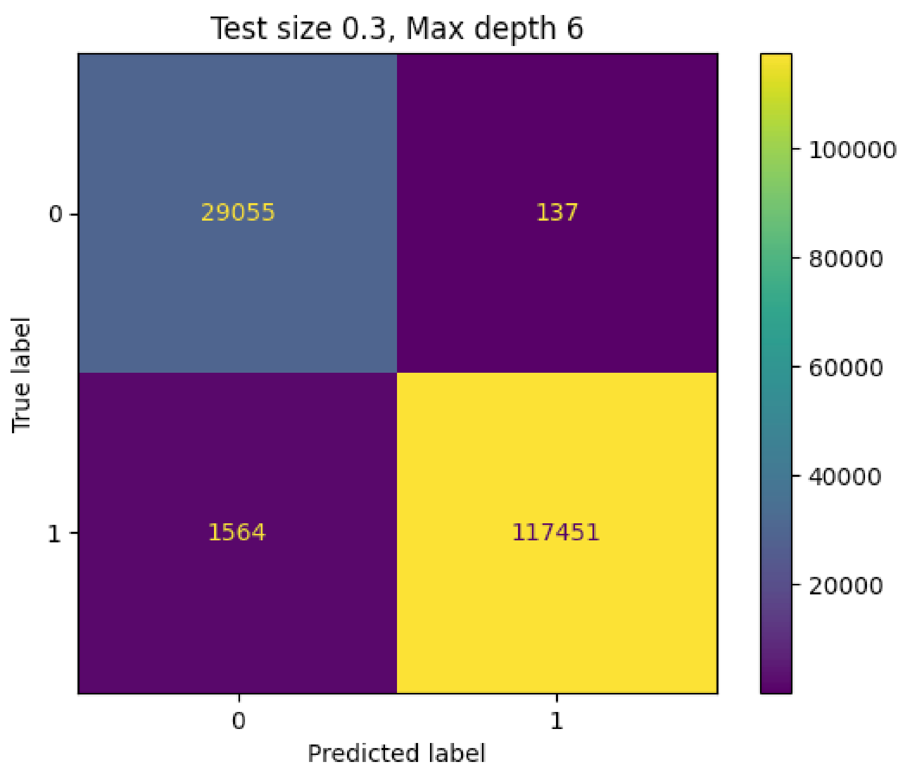
-compare the classification performance of tuned Decision Tree

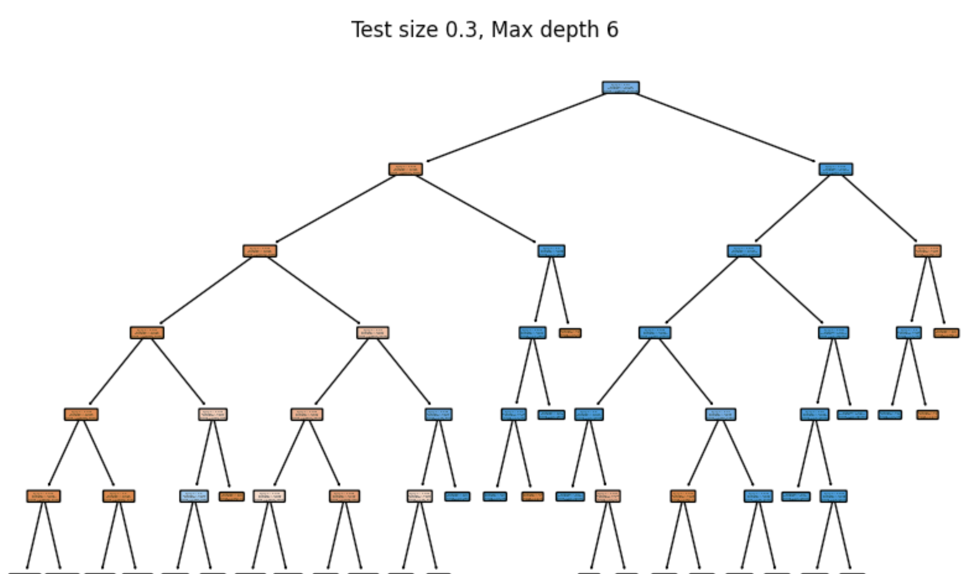
```
Test size 0.3, max depth 4: accuracy = 0.9857226716686797
Classification report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.96     29192
           1       1.00      0.98      0.99    119015

    accuracy                           0.99    148207
   macro avg       0.97      0.99      0.98    148207
weighted avg       0.99      0.99      0.99    148207
```



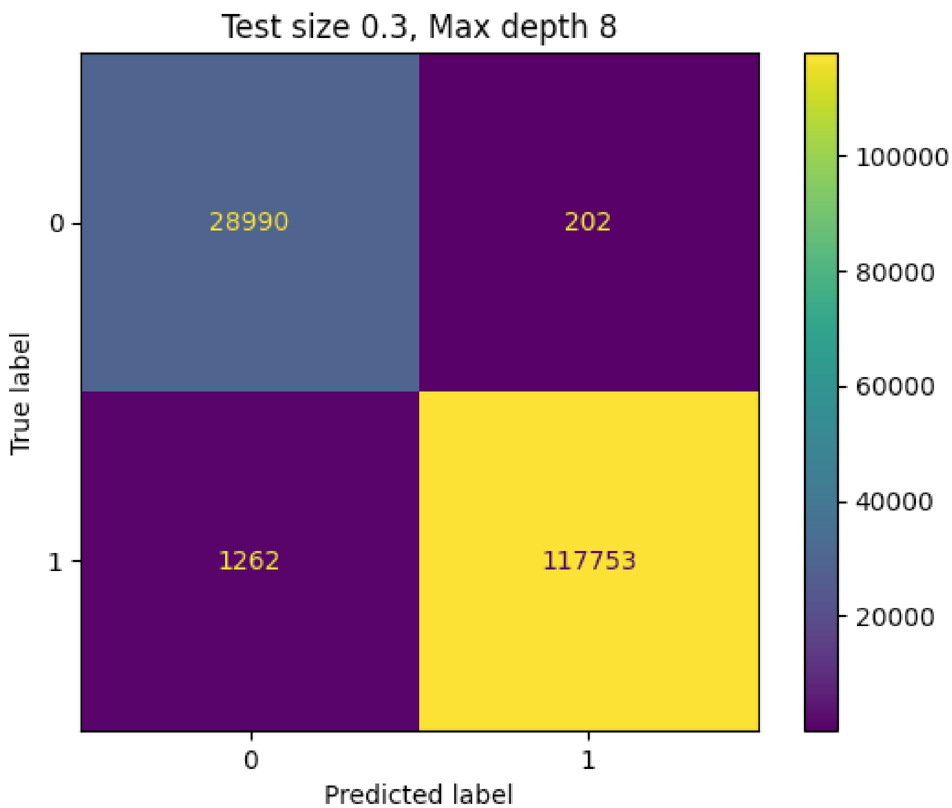


```
Test size 0.3, max depth 6: accuracy = 0.988522809314
Classification report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97     29192
           1       1.00      0.99      0.99    119015

    accuracy                           0.99    148207
   macro avg       0.97      0.99      0.98    148207
weighted avg       0.99      0.99      0.99    148207
```

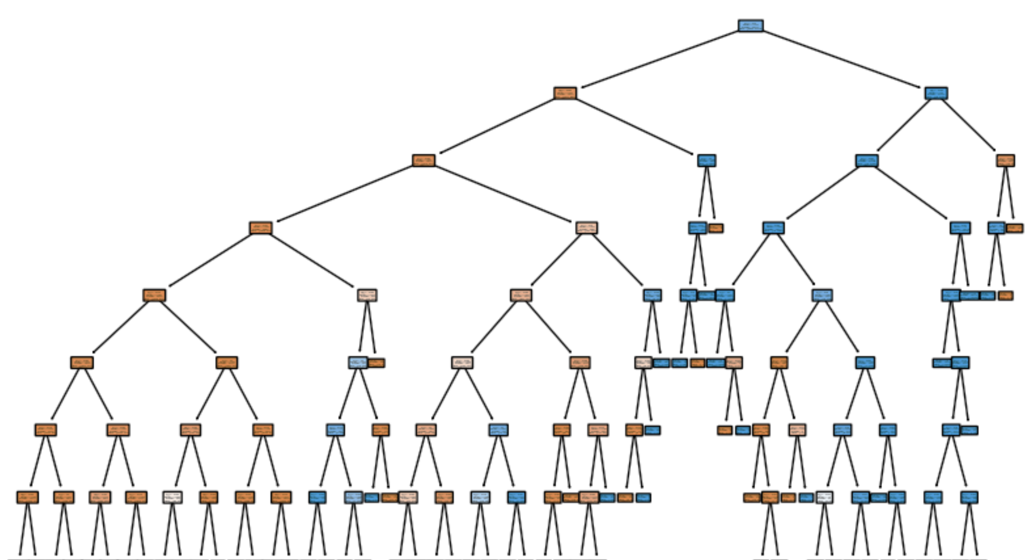




```
Test size 0.3, max depth 8: accuracy = 0.9901219240656649
Classification report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     29192
           1       1.00      0.99      0.99    119015

    accuracy                           0.99    148207
   macro avg       0.98      0.99      0.98    148207
weighted avg       0.99      0.99      0.99    148207
```



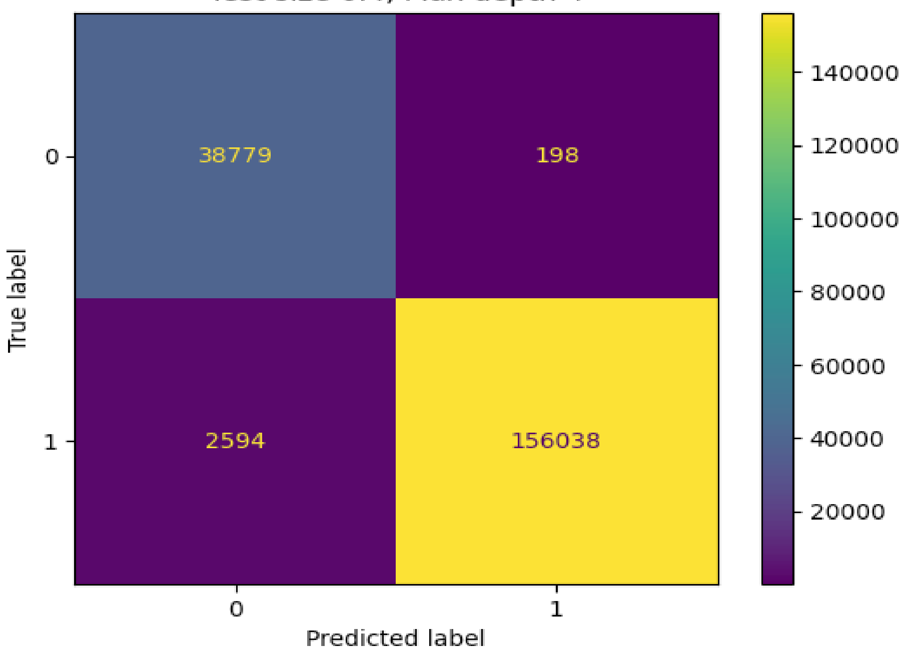
Test size 0.3, Max depth 8



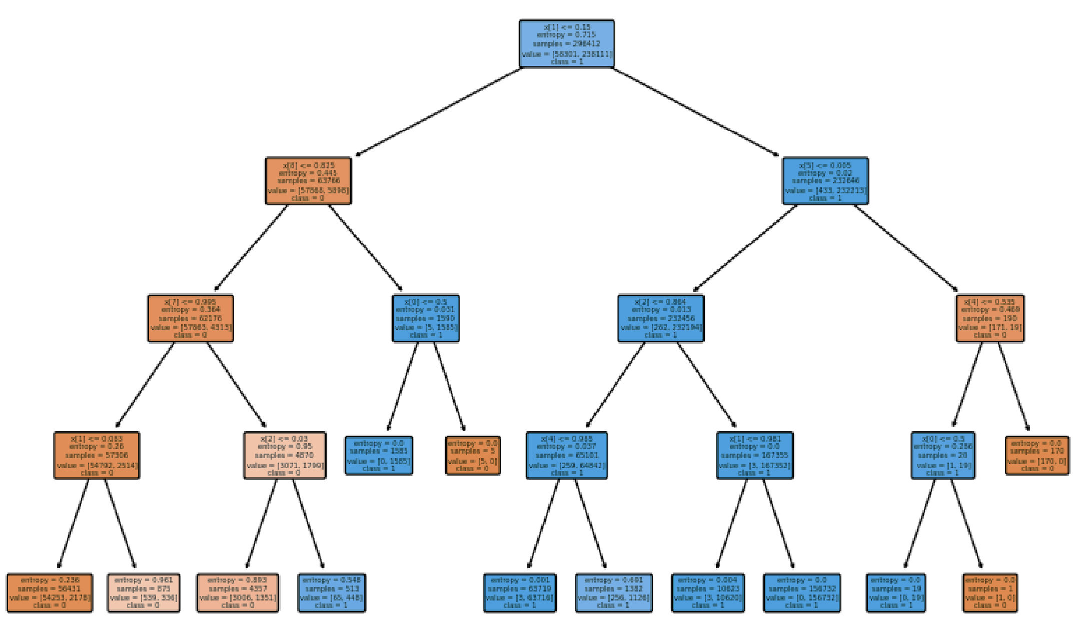
Best split for Test size 0.3 (Entropy): max depth = 8, accuracy = 0.9901219240656649

```
Test size 0.4, max depth 4: accuracy = 0.9858710888674099
Classification report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.97     38977
           1       1.00      0.98      0.99    158632

    accuracy                           0.99    197609
   macro avg       0.97      0.99      0.98    197609
weighted avg       0.99      0.99      0.99    197609
```

Test size 0.4, Max depth 4

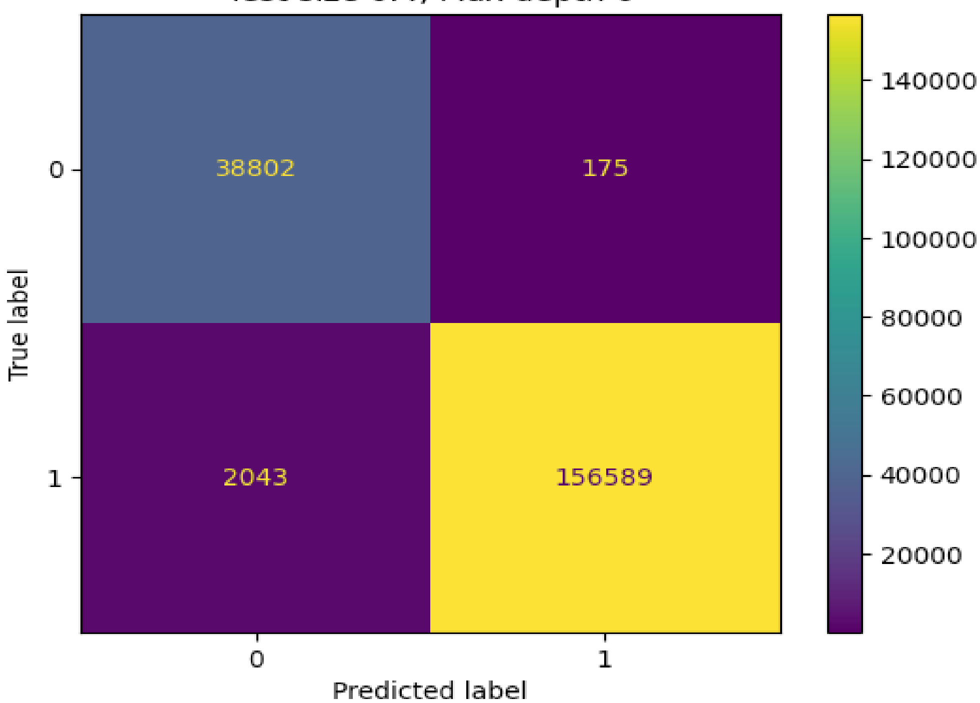


Test size 0.4, Max depth 4

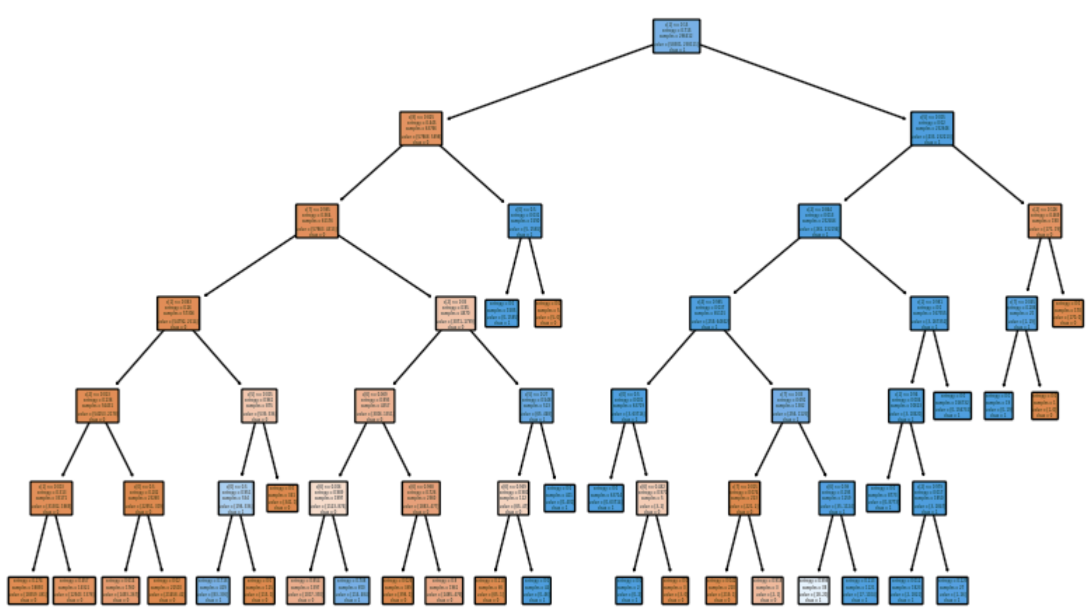


```
Test size 0.4, max depth 6: accuracy = 0.9887758148667318
Classification report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97     38977
           1       1.00      0.99      0.99    158632

    accuracy                           0.99    197609
   macro avg       0.97      0.99      0.98    197609
weighted avg       0.99      0.99      0.99    197609
```

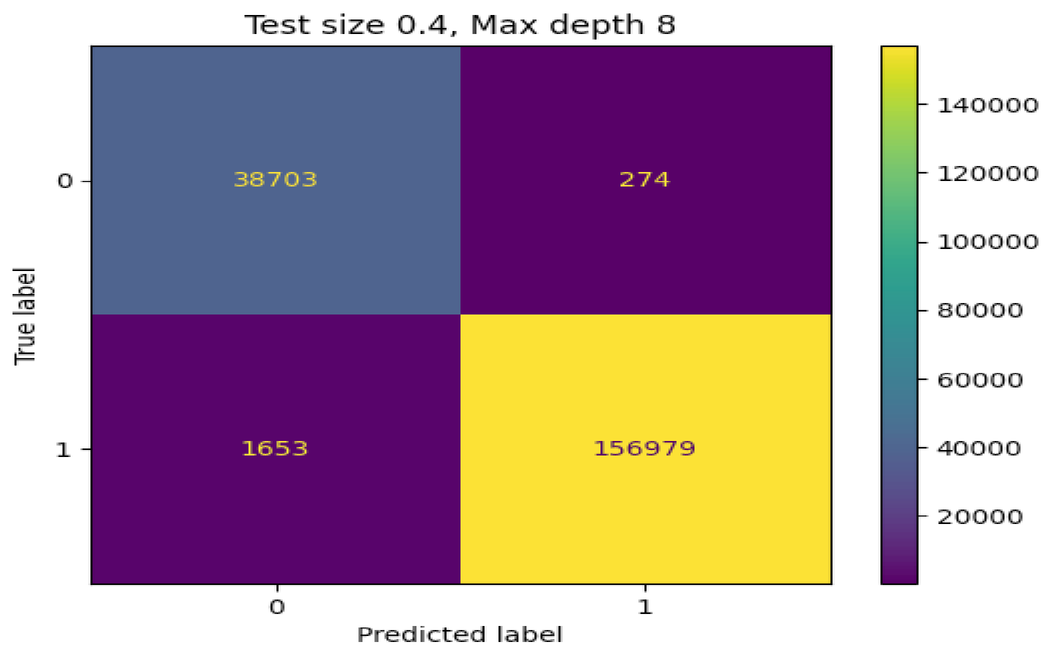
Test size 0.4, Max depth 6

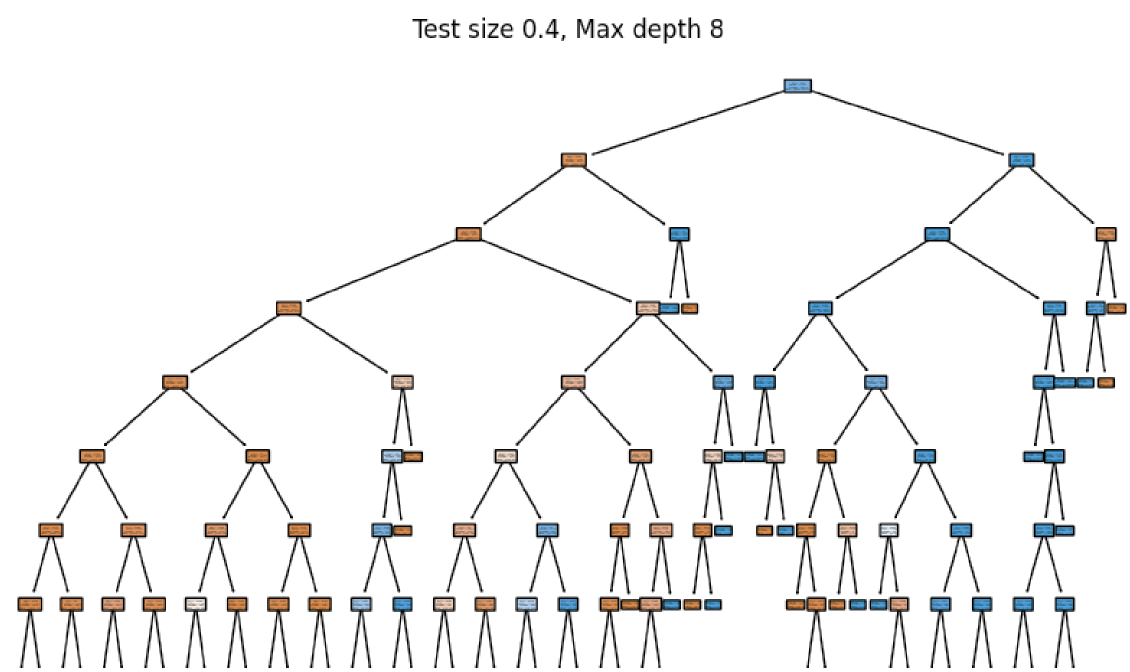


Test size 0.4, Max depth 6



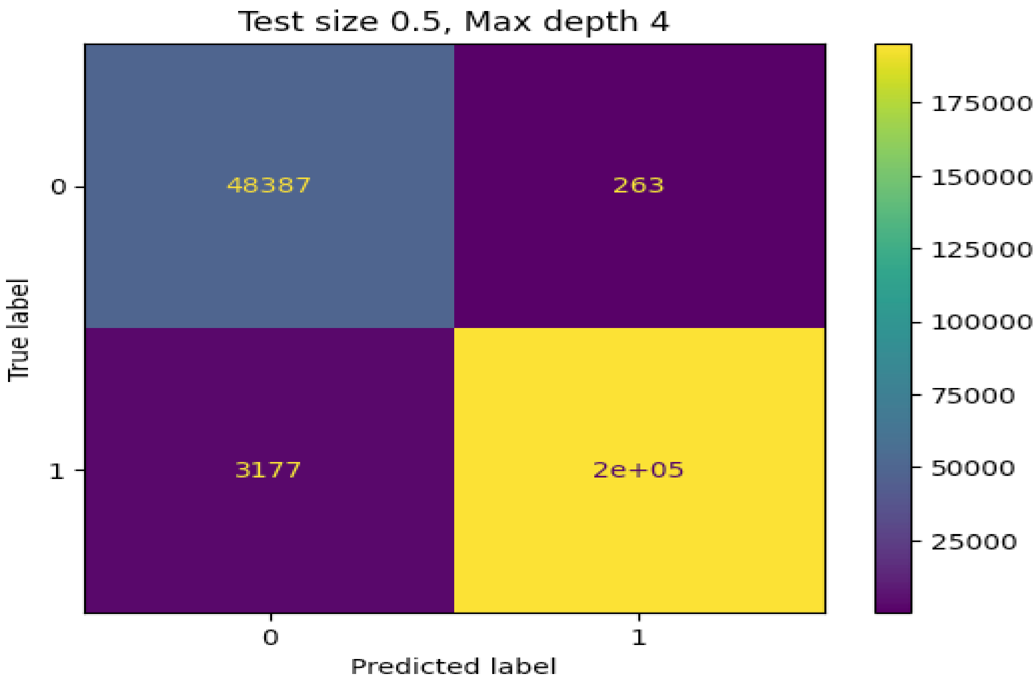
```
Test size 0.4, max depth 8: accuracy = 0.9902484198594194
Classification report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     38977
           1       1.00      0.99      0.99    158632

    accuracy                           0.99    197609
   macro avg       0.98      0.99      0.98    197609
weighted avg       0.99      0.99      0.99    197609
```

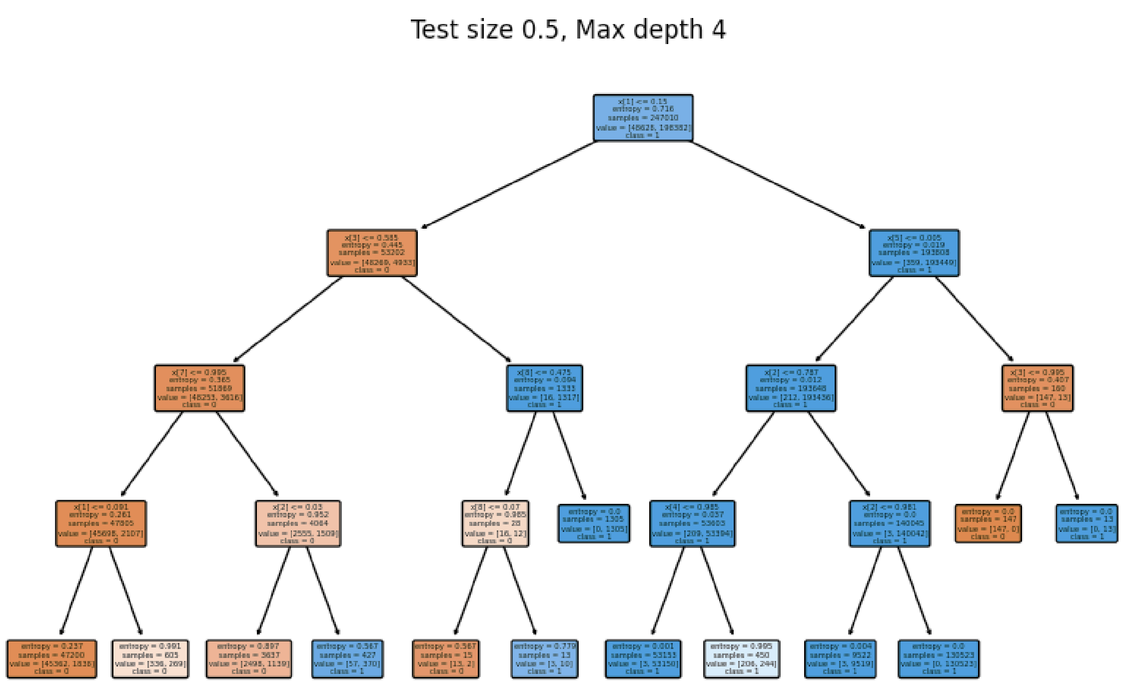




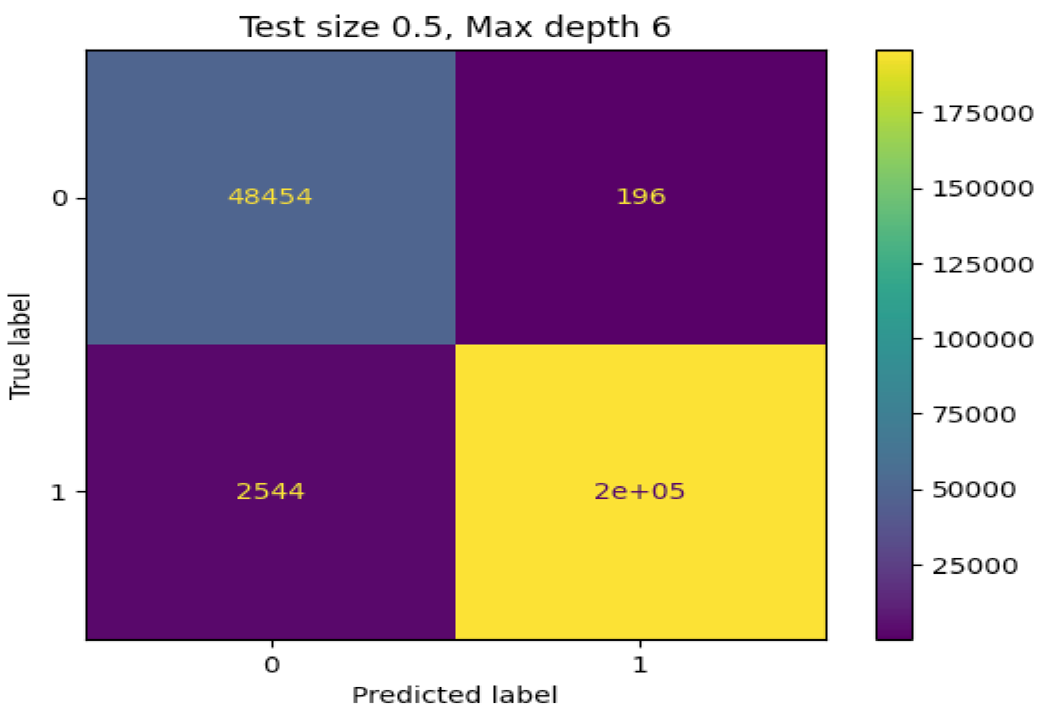

```
Best split for Test size 0.4 (Entropy): max depth = 8, accuracy = 0.9902484198594194
```

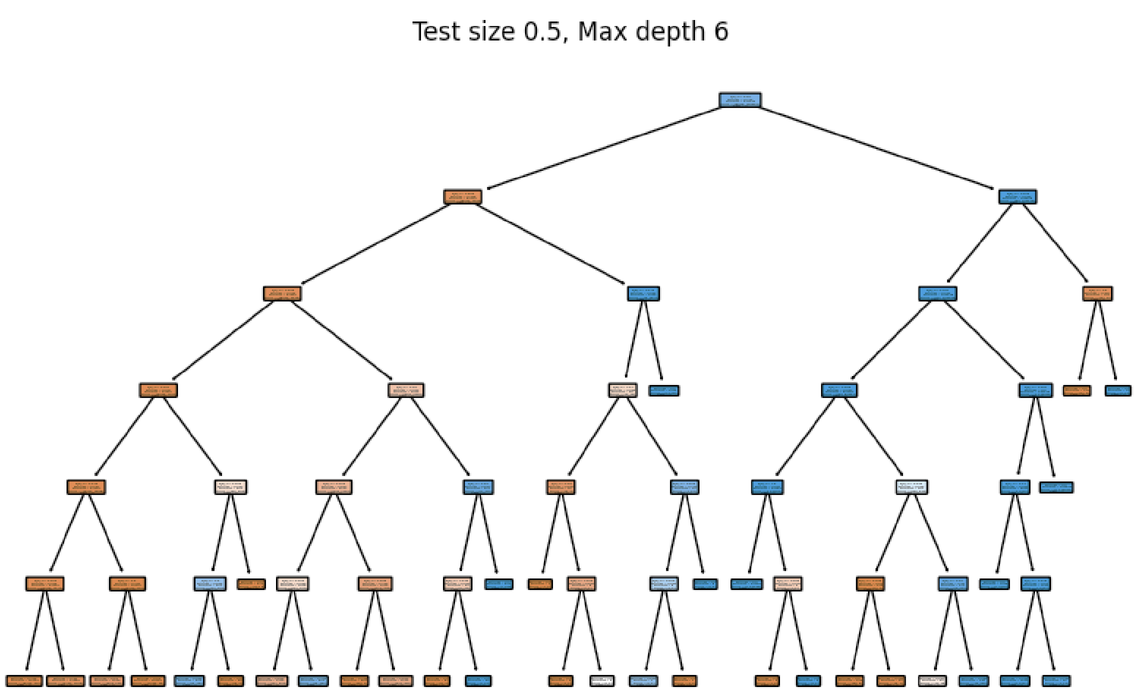
```
Test size 0.5, max depth 4: accuracy = 0.986073494702665
Classification report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.97     48650
           1       1.00      0.98      0.99    198361

    accuracy                           0.99    247011
   macro avg       0.97      0.99      0.98    247011
weighted avg       0.99      0.99      0.99    247011
```



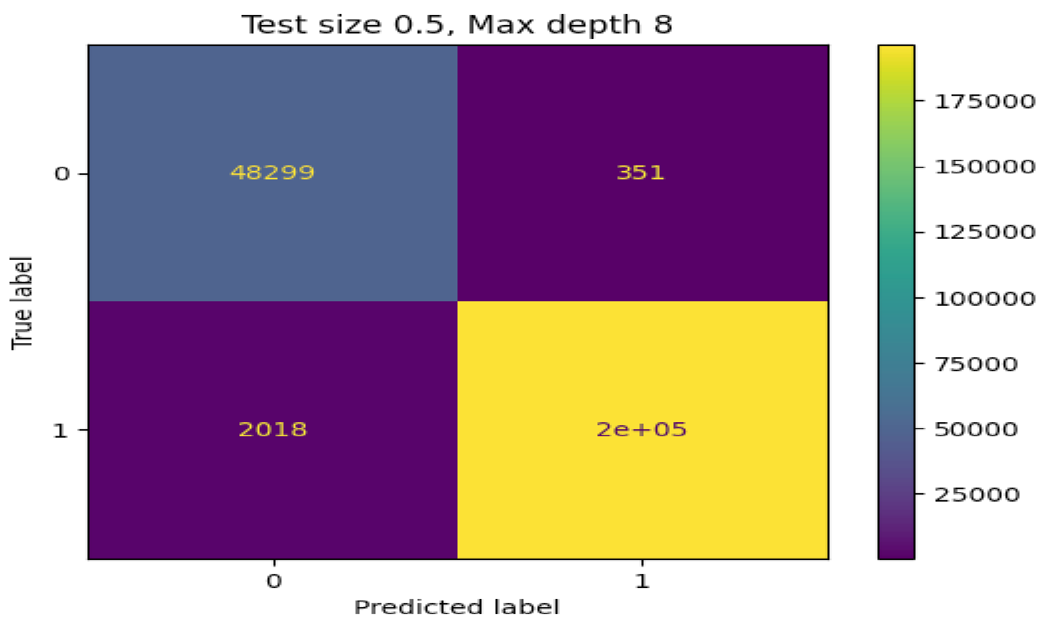


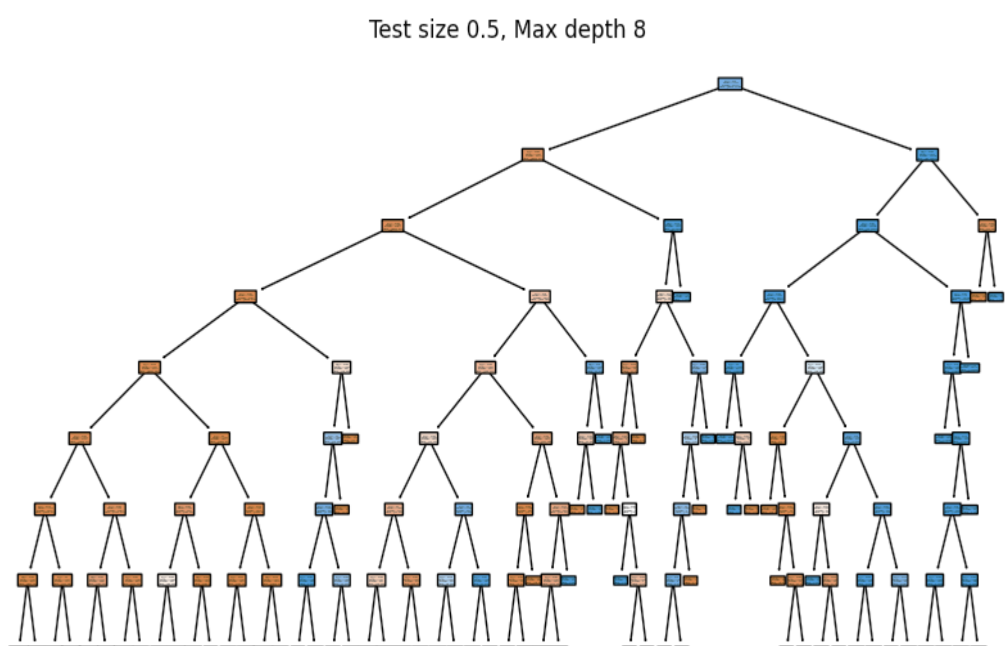
```
Test size 0.5, max depth 6: accuracy = 0.9889073765945646
Classification report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97     48650
           1       1.00      0.99      0.99    198361

    accuracy                           0.99    247011
   macro avg       0.97      0.99      0.98    247011
weighted avg       0.99      0.99      0.99    247011
```





```
Test size 0.5, max depth 8: accuracy = 0.9904093339972714
Classification report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     48650
           1       1.00      0.99      0.99    198361

    accuracy                           0.99    247011
   macro avg       0.98      0.99      0.99    247011
weighted avg       0.99      0.99      0.99    247011
```






```
Best split for Test size 0.5 (Entropy): max depth = 8, accuracy = 0.9904093339972714
```

Step 5: Mitigation Strategies

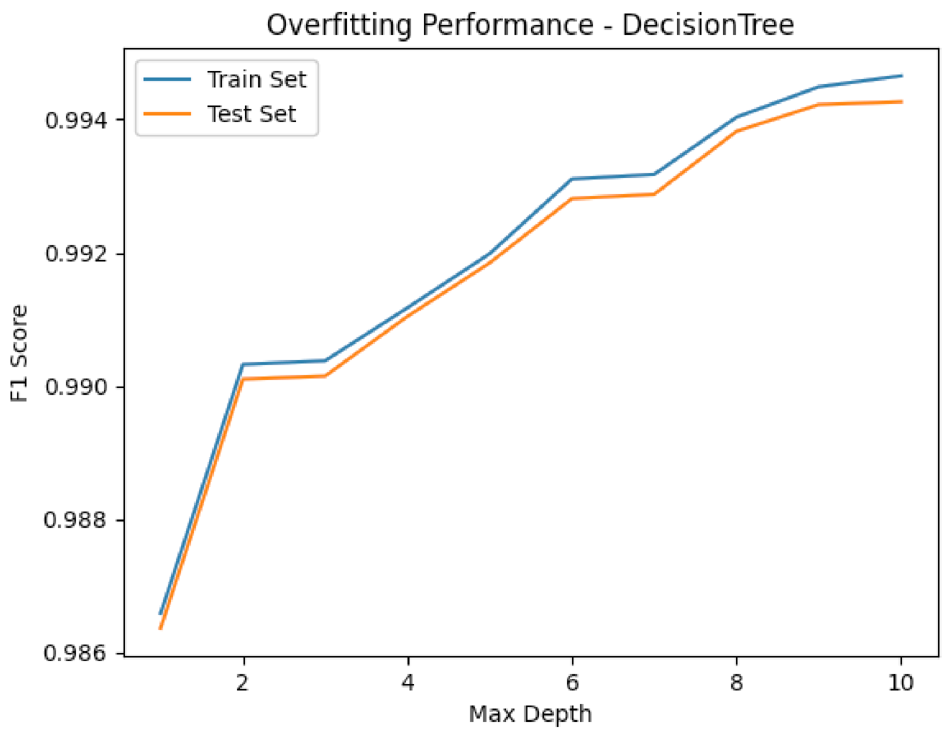
- Three mitigation strategies are applied to the decision tree classifier to improve its performance:

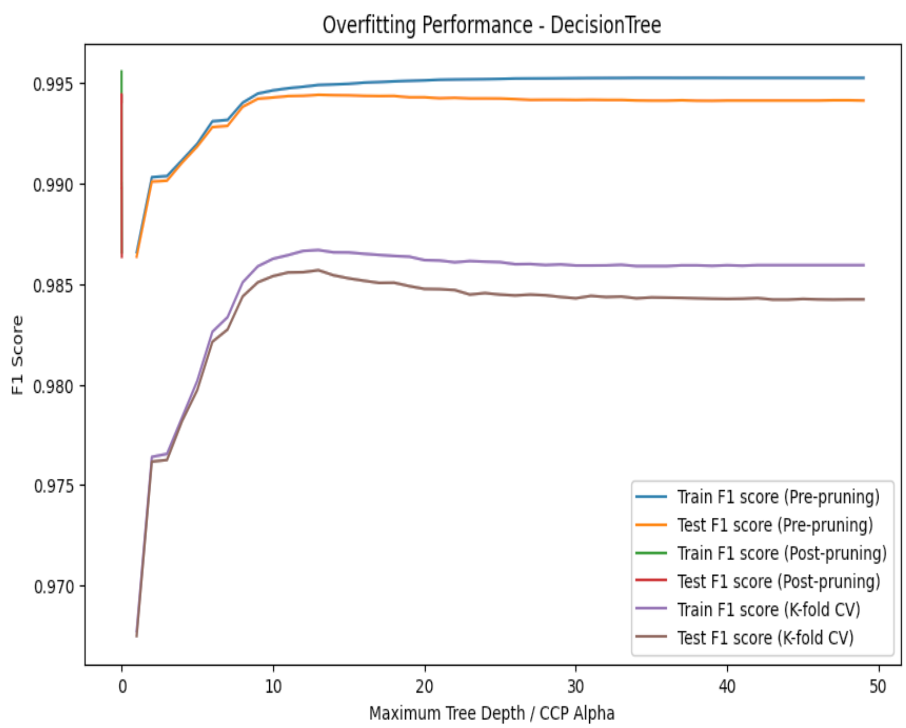
- Pre-pruning: Controlling the tree's growth during the training process by setting parameters like maximum depth or minimum samples required to split.

- Post-pruning: Pruning the fully grown tree by removing unnecessary branches based on their impact on validation data.

- K-fold Cross-validation: Evaluating the classifier's performance using k-fold cross-validation to assess its generalization ability.

```
Subset 1:
Number of training examples: 345814
Number of test examples: 148207
Training labels distribution: (array([0, 1]), array([ 68086, 277728]))
Test labels distribution: (array([0, 1]), array([ 29192, 119015]))

DecisionTree without mitigation strategies:
F1 score on train data: 0.9946391603460589
F1 score on test data: 0.9942862447884139

DecisionTree with pre-pruning:
F1 score on train data: 0.9919838988966755
F1 score on test data: 0.9918455607605168

DecisionTree with post-pruning:
F1 score on train data: 0.9955735177429821
F1 score on test data: 0.9940742552751781

DecisionTree with k-fold cross-validation:
F1 score on train data: 0.986 +/- 0.001
F1 score on test data: 0.984 +/- 0.001
```


Overfitting Performance - DecisionTree
Train Set
Test Set
0.994
0.992
0.990
0.988
0.986
F1 Score
2
4
6
8
10
Max Depth


Overfitting Performance - DecisionTree
0.995
0.990
0.985
0.980
0.975
0.970
F1 Score
0
10
20
30
40
50
Maximum Tree Depth / CCP Alpha
Train F1 score (Pre-pruning)
Test F1 score (Pre-pruning)
Train F1 score (Post-pruning)
Test F1 score (Post-pruning)
Train F1 score (K-fold CV)
Test F1 score (K-fold CV)