In [36]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, classification_report
```

In [2]:
```
df = pd.read_csv("./bank.csv", delimiter=";")
```

In [3]:
```
df.head()
```

Out[3]:

	age	job	marital	education	default	balance	housing	loan	contact
0	30	unemployed	married	primary	no	1787	no	no	cellular
1	33	services	married	secondary	no	4789	yes	yes	cellular
2	35	management	single	tertiary	no	1350	yes	no	cellular
3	30	management	married	tertiary	no	1476	yes	yes	unknown
4	59	blue-collar	married	secondary	no	0	yes	no	unknown

In [4]:
```
df.describe()
```

Out[4]:

	age	balance	day	duration	campaign	pdays
count	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000
mean	41.170095	1422.657819	15.915284	263.961292	2.793630	39.766645
std	10.576211	3009.638142	8.247667	259.856633	3.109807	100.121124
min	19.000000	-3313.000000	1.000000	4.000000	1.000000	-1.000000
25%	33.000000	69.000000	9.000000	104.000000	1.000000	-1.000000
50%	39.000000	444.000000	16.000000	185.000000	2.000000	-1.000000
75%	49.000000	1480.000000	21.000000	329.000000	3.000000	-1.000000
max	87.000000	71188.000000	31.000000	3025.000000	50.000000	871.000000

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4521 entries, 0 to 4520
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        4521 non-null   int64 
 1   job        4521 non-null   object
 2   marital    4521 non-null   object
 3   education  4521 non-null   object
 4   default    4521 non-null   object
 5   balance    4521 non-null   int64 
 6   housing    4521 non-null   object
 7   loan       4521 non-null   object
 8   contact    4521 non-null   object
 9   day        4521 non-null   int64 
 10  month      4521 non-null   object
 11  duration   4521 non-null   int64 
 12  campaign   4521 non-null   int64 
 13  pdays      4521 non-null   int64 
 14  previous   4521 non-null   int64 
 15  poutcome   4521 non-null   object
 16  y          4521 non-null   object
dtypes: int64(7), object(10)
memory usage: 600.6+ KB
```

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y            0
dtype: int64
```

In [7]:
```
df['marital'].describe()
```

Out[7]:
```
count        4521
unique          3
top       married
freq         2797
Name: marital, dtype: object
```

In [8]:
```
df['marital'].unique()
```

```
Out[8]: array(['married', 'single', 'divorced'], dtype=object)
```

```
In [9]: le_marital = LabelEncoder()
        df['marital'] = le_marital.fit_transform(df['marital'])
        print(le_marital.classes_)
        df['marital']
```

```
['divorced' 'married' 'single']
```

```
Out[9]: 0       1
        1       1
        2       2
        3       1
        4       1
               ..
        4516    1
        4517    1
        4518    1
        4519    1
        4520    2
        Name: marital, Length: 4521, dtype: int64
```

```
In [10]: df['job'].unique()
```

```
Out[10]: array(['unemployed', 'services', 'management', 'blue-collar',
                'self-employed', 'technician', 'entrepreneur', 'admin.', 'student',
                'housemaid', 'retired', 'unknown'], dtype=object)
```

```
In [11]: le_job = LabelEncoder()
         df['job'] = le_job.fit_transform(df['job'])
         print(le_job.classes_)
         df['job']
```

```
['admin.' 'blue-collar' 'entrepreneur' 'housemaid' 'management' 'retired'
 'self-employed' 'services' 'student' 'technician' 'unemployed' 'unknown']
```

```
Out[11]: 0       10
         1        7
         2        4
         3        4
         4        1
                 ..
         4516     7
         4517     6
         4518     9
         4519     1
         4520     2
         Name: job, Length: 4521, dtype: int64
```

```
In [12]: df['education'].unique()
```

```
Out[12]: array(['primary', 'secondary', 'tertiary', 'unknown'], dtype=object)
```

```
In [13]: le_education = LabelEncoder()
         df['education'] = le_education.fit_transform(df['education'])
         print(le_education.classes_)
         df['education']
```

```
['primary' 'secondary' 'tertiary' 'unknown']
```

```
Out[13]: 0       0
         1       1
         2       2
         3       2
         4       1
                ..
         4516    1
         4517    2
         4518    1
         4519    1
         4520    2
         Name: education, Length: 4521, dtype: int64
```

```
In [14]: df['default'].unique()
```

```
Out[14]: array(['no', 'yes'], dtype=object)
```

```
In [15]: df['default'].replace(['no', 'yes'], [0, 1], inplace=True)
         print(df['default'])
```

```
0       0
1       0
2       0
3       0
4       0
       ..
4516    0
4517    1
4518    0
4519    0
4520    0
Name: default, Length: 4521, dtype: int64
```

```
In [16]: df['housing'].unique()
```

```
Out[16]: array(['no', 'yes'], dtype=object)
```

```
In [17]: df['housing'].replace(['no', 'yes'], [0, 1], inplace=True)
         print(df['housing'])
```

```
0       0
1       1
2       1
3       1
4       1
       ..
4516    1
4517    1
4518    0
4519    0
4520    1
Name: housing, Length: 4521, dtype: int64
```

```
In [18]: df['loan'].unique()
```

```
Out[18]: array(['no', 'yes'], dtype=object)
```

```
In [19]: df['loan'].replace(['no', 'yes'], [0, 1], inplace=True)
         print(df['loan'])
```

```
0       0
1       1
2       0
3       1
4       0
       ..
4516    0
4517    1
4518    0
4519    0
4520    1
Name: loan, Length: 4521, dtype: int64
```

```
In [20]: df['contact'].unique()
```

```
Out[20]: array(['cellular', 'unknown', 'telephone'], dtype=object)
```

```
In [21]: le_contact = LabelEncoder()
         df['contact'] = le_contact.fit_transform(df['contact'])
         print(le_contact.classes_)
         df['contact']
```

```
['cellular' 'telephone' 'unknown']
```

```
Out[21]: 0       0
         1       0
         2       0
         3       2
         4       2
                ..
         4516    0
         4517    2
         4518    0
         4519    0
         4520    0
         Name: contact, Length: 4521, dtype: int64
```

```
In [22]: df['month'].unique()
```

```
Out[22]: array(['oct', 'may', 'apr', 'jun', 'feb', 'aug', 'jan', 'jul', 'nov',
                'sep', 'mar', 'dec'], dtype=object)
```

```
In [23]: le_month = LabelEncoder()
         df['month'] = le_month.fit_transform(df['month'])
         print(le_month.classes_)
         df['month']
```

```
['apr' 'aug' 'dec' 'feb' 'jan' 'jul' 'jun' 'mar' 'may' 'nov' 'oct' 'sep']
```

Out[23]:
```
0       10
1        8
2        0
3        6
4        8
        ..
4516     5
4517     8
4518     1
4519     3
4520     0
Name: month, Length: 4521, dtype: int64
```

In [24]:
```
df['poutcome'].unique()
```

Out[24]:
```
array(['unknown', 'failure', 'other', 'success'], dtype=object)
```

In [25]:
```
le_poutcome = LabelEncoder()
df['poutcome'] = le_poutcome.fit_transform(df['poutcome'])
print(le_poutcome.classes_)
df['poutcome']
```

```
['failure' 'other' 'success' 'unknown']
```

Out[25]:
```
0       3
1       0
2       0
3       3
4       3
       ..
4516    3
4517    3
4518    3
4519    1
4520    1
Name: poutcome, Length: 4521, dtype: int64
```

In [26]:
```
df.head()
```

Out[26]:

	age	job	marital	education	default	balance	housing	loan	contact	day	mont
0	30	10	1	0	0	1787	0	0	0	19	1
1	33	7	1	1	0	4789	1	1	0	11	
2	35	4	2	2	0	1350	1	0	0	16	
3	30	4	1	2	0	1476	1	1	2	3	
4	59	1	1	1	0	0	1	0	2	5	

In [27]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4521 entries, 0 to 4520
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        4521 non-null   int64 
 1   job        4521 non-null   int64 
 2   marital    4521 non-null   int64 
 3   education  4521 non-null   int64 
 4   default    4521 non-null   int64 
 5   balance    4521 non-null   int64 
 6   housing    4521 non-null   int64 
 7   loan       4521 non-null   int64 
 8   contact    4521 non-null   int64 
 9   day        4521 non-null   int64 
 10  month      4521 non-null   int64 
 11  duration   4521 non-null   int64 
 12  campaign   4521 non-null   int64 
 13  pdays      4521 non-null   int64 
 14  previous   4521 non-null   int64 
 15  poutcome   4521 non-null   int64 
 16  y          4521 non-null   object
dtypes: int64(16), object(1)
memory usage: 600.6+ KB
```

In [28]:
```
df.shape
```

Out[28]:
```
(4521, 17)
```

In [29]:
```
X = df.iloc[:, :-1].values
y = df.iloc[:, -1].values
```

In [30]:
```
X.shape, y.shape
```

Out[30]:
```
((4521, 16), (4521,))
```

In [31]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1234)
```

In [32]:
```
print(f"{X_train.shape=}\n{y_train.shape=}\n\n{X_test.shape=}\n{y_test.shape
```

```
X_train.shape=(3390, 16)
y_train.shape=(3390,)

X_test.shape=(1131, 16)
y_test.shape=(1131,)
```

In [33]:
```
dtc = DecisionTreeClassifier()
```

In [34]:
```
dtc.fit(X_train, y_train)
```

Out[34]:
```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

In [40]:
```
y_pred_test = dtc.predict(X_test)
y_pred_train = dtc.predict(X_train)
```

In [42]:
```
print(classification_report(y_train, y_pred_train))
```

```
              precision    recall  f1-score   support

          no       1.00      1.00      1.00      3003
         yes       1.00      1.00      1.00       387

    accuracy                           1.00      3390
   macro avg       1.00      1.00      1.00      3390
weighted avg       1.00      1.00      1.00      3390
```

In [47]:
```
confusion_matrix(y_train, y_pred_train)
```

Out[47]:
```
array([[3003,    0],
       [   0,  387]])
```

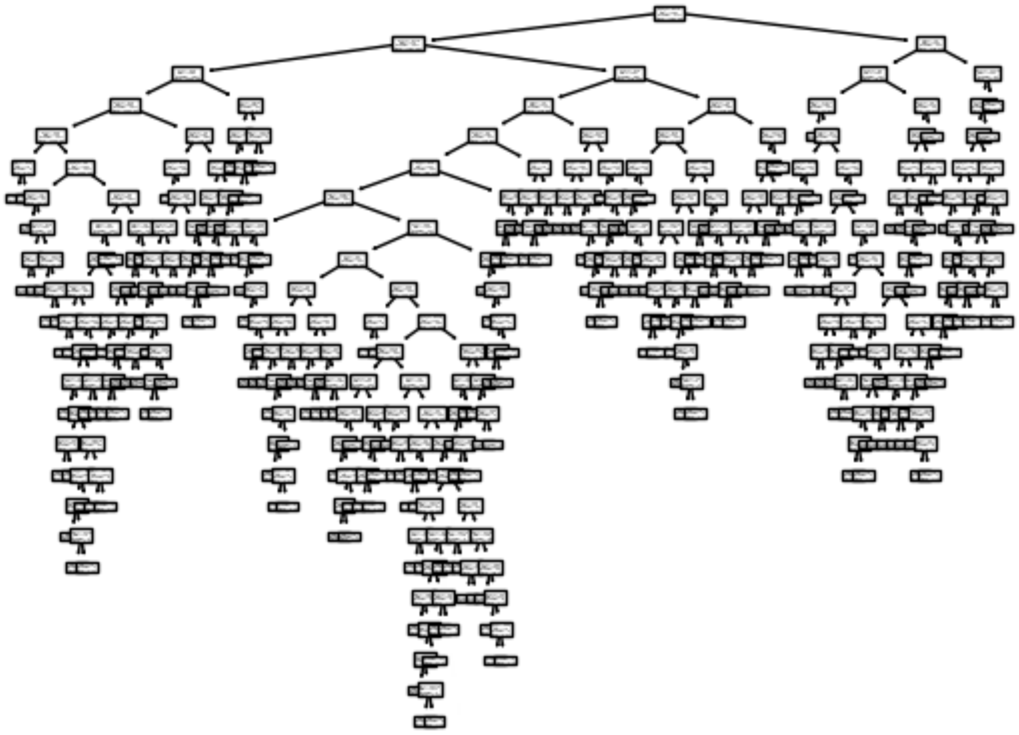
In [45]:
```
print(classification_report(y_test, y_pred_test))
```

```
              precision    recall  f1-score   support

          no       0.92      0.92      0.92       997
         yes       0.37      0.37      0.37       134

    accuracy                           0.85      1131
   macro avg       0.64      0.64      0.64      1131
weighted avg       0.85      0.85      0.85      1131
```

In [46]:
```
confusion_matrix(y_test, y_pred_test)
```

Out[46]:
```
array([[913,  84],
       [ 84,  50]])
```

In [54]:
```
import matplotlib.pyplot as plt

plot_tree(dtc)
plt.show()
plt.savefig("as")
```

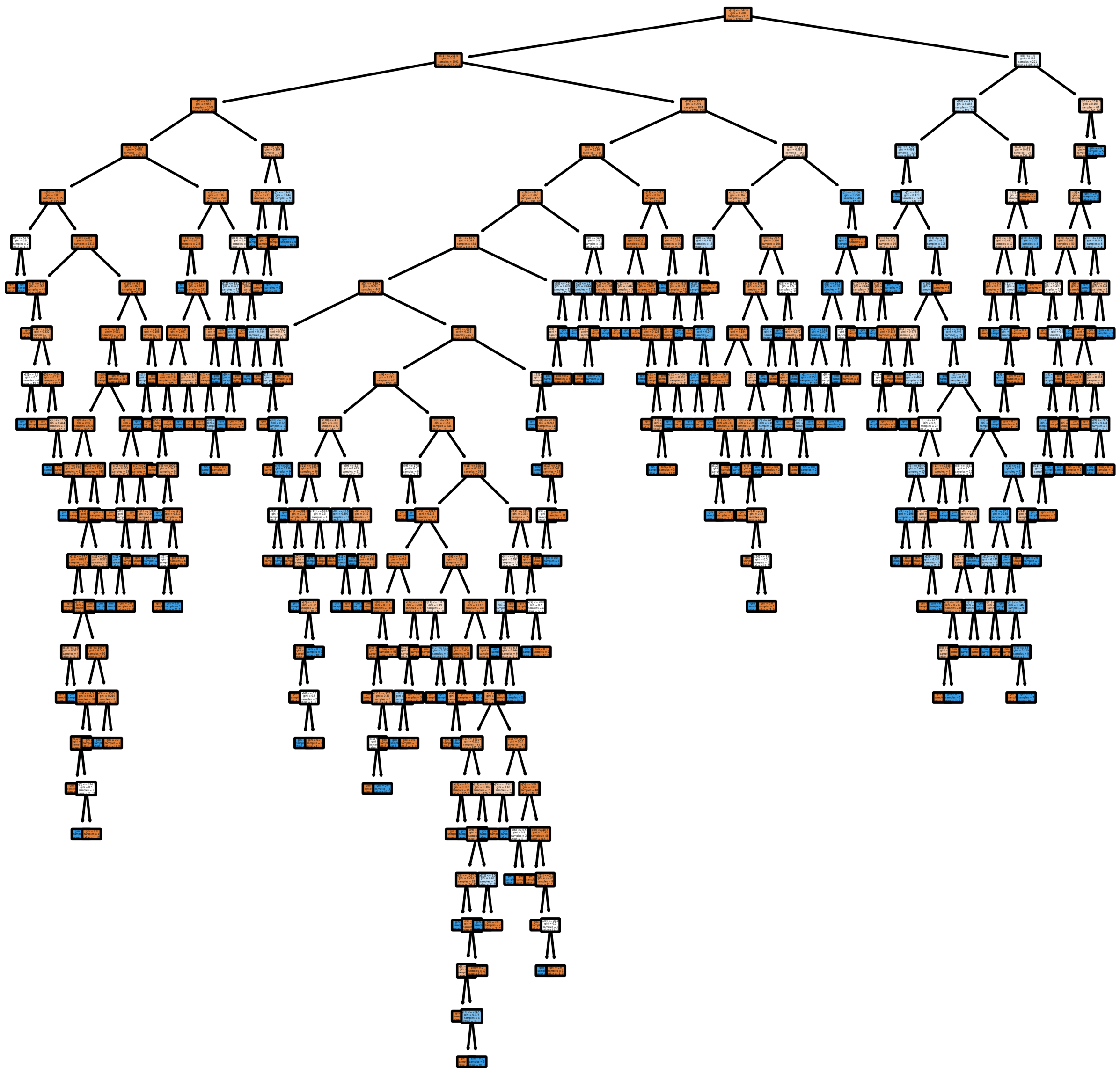


```
<Figure size 640x480 with 0 Axes>
```

In [53]:
```
dtc.get_depth(), dtc.get_n_leaves()
```

Out[53]:
```
(23, 239)
```

In [56]:
```
fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(8, 8), dpi=600)
plot_tree(dtc, filled=True, rounded=True, ax=axes)
plt.savefig('tree.svg')
```



```
In [57]: dtc_1 = DecisionTreeClassifier(max_depth=5)
```

```
In [58]: dtc_1.fit(X_train, y_train)
```

```
Out[58]: ▾       DecisionTreeClassifier

         DecisionTreeClassifier(max_depth=5)
```

```
In [60]: y_pred_test = dtc_1.predict(X_test)
         y_pred_train = dtc_1.predict(X_train)
```

```
In [61]: print(classification_report(y_test, y_pred_test))
```

```
              precision    recall  f1-score   support

          no       0.92      0.97      0.94      1608
         yes       0.53      0.32      0.40       201

    accuracy                           0.89      1809
   macro avg       0.73      0.64      0.67      1809
weighted avg       0.88      0.89      0.88      1809
```

```
In [62]: confusion_matrix(y_test, y_pred_test)
```

```
Out[62]: array([[1552,   56],
                [ 137,   64]])
```

```
In [64]: print(classification_report(y_train, y_pred_train))
```

```
              precision    recall  f1-score   support

          no       0.93      0.98      0.95      2392
         yes       0.75      0.47      0.58       320

    accuracy                           0.92      2712
   macro avg       0.84      0.72      0.77      2712
weighted avg       0.91      0.92      0.91      2712
```

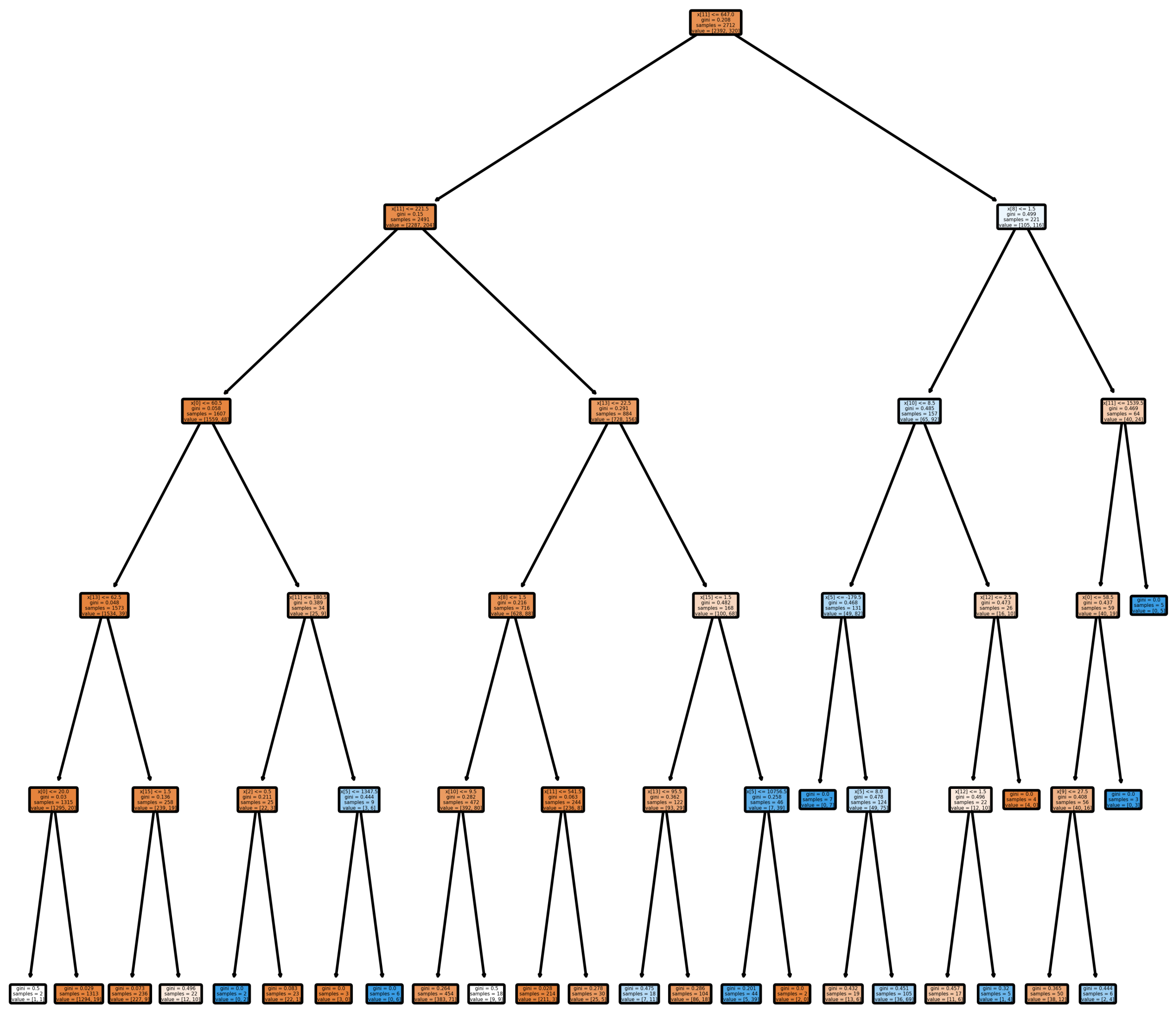
```
In [65]: confusion_matrix(y_train, y_pred_train)
```

```
Out[65]: array([[2341,   51],
                [ 170,  150]])
```

```
In [67]: dtc_1.get_depth(), dtc_1.get_n_leaves()
```

```
Out[67]: (5, 26)
```

```
In [69]: fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(8, 8), dpi=600)
         plot_tree(dtc_1, filled=True, rounded=True, ax=axes)
         plt.savefig('tree_1.svg')
```


x[11] <= 647.0 gini = 0.208 samples = 2712 value = [2392, 320]
x[11] <= 221.5 gini = 0.15 samples = 2491 value = [2287, 204]
x[8] <= 1.5 gini = 0.499 samples = 221 value = [105, 116]
x[0] <= 60.5 gini = 0.058 samples = 1607 value = [1559, 48]
x[13] <= 22.5 gini = 0.291 samples = 884 value = [728, 156]
x[10] <= 8.5 gini = 0.485 samples = 157 value = [65, 92]
x[11] <= 1539.5 gini = 0.469 samples = 64 value = [40, 24]
x[13] <= 62.5 gini = 0.048 samples = 1573 value = [1534, 39]
x[11] <= 180.5 gini = 0.389 samples = 34 value = [25, 9]
x[8] <= 1.5 gini = 0.216 samples = 716 value = [628, 88]
x[15] <= 1.5 gini = 0.482 samples = 168 value = [100, 68]
x[5] <= -179.5 gini = 0.468 samples = 131 value = [49, 82]
x[12] <= 2.5 gini = 0.473 samples = 26 value = [16, 10]
x[0] <= 58.5 gini = 0.437 samples = 59 value = [40, 19]
gini = 0.0 samples = 5 value = [0, 5]
x[0] <= 20.0 gini = 0.03 samples = 1315 value = [1295, 20]
x[15] <= 1.5 gini = 0.136 samples = 258 value = [239, 19]
x[2] <= 0.5 gini = 0.211 samples = 25 value = [22, 3]
x[5] <= 1347.5 gini = 0.444 samples = 9 value = [3, 6]
x[10] <= 9.5 gini = 0.282 samples = 472 value = [392, 80]
x[11] <= 541.5 gini = 0.063 samples = 244 value = [236, 8]
x[13] <= 95.5 gini = 0.362 samples = 122 value = [93, 29]
x[5] <= 10756.5 gini = 0.258 samples = 46 value = [7, 39]
gini = 0.0 samples = 7 value = [0, 7]
x[5] <= 8.0 gini = 0.478 samples = 124 value = [49, 75]
x[12] <= 1.5 gini = 0.496 samples = 22 value = [12, 10]
gini = 0.0 samples = 4 value = [4, 0]
x[9] <= 27.5 gini = 0.408 samples = 56 value = [40, 16]
gini = 0.0 samples = 3 value = [0, 3]
gini = 0.5 samples = 2 value = [1, 1]
gini = 0.029 samples = 1313 value = [1294, 19]
gini = 0.073 samples = 236 value = [227, 9]
gini = 0.496 samples = 22 value = [12, 10]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.083 samples = 23 value = [22, 1]
gini = 0.0 samples = 3 value = [3, 0]
gini = 0.0 samples = 6 value = [0, 6]
gini = 0.264 samples = 454 value = [383, 71]
gini = 0.5 samples = 18 value = [9, 9]
gini = 0.028 samples = 214 value = [211, 3]
gini = 0.278 samples = 30 value = [25, 5]
gini = 0.475 samples = 18 value = [7, 11]
gini = 0.286 samples = 104 value = [86, 18]
gini = 0.201 samples = 44 value = [5, 39]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.432 samples = 19 value = [13, 6]
gini = 0.451 samples = 105 value = [36, 69]
gini = 0.457 samples = 17 value = [11, 6]
gini = 0.32 samples = 5 value = [1, 4]
gini = 0.365 samples = 50 value = [38, 12]
gini = 0.444 samples = 6 value = [2, 4]

```
In [ ]:
```