In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
from sklearn.metrics import mean_squared_error
```

In [3]:
```
df = pd.read_csv("D:\\AirQualityUCI.csv")
df
```

Out[3]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.
0	10-03-2004	18:00:00	2.6	1360.0	150.0	11.9	1046.0	166.0	
1	10-03-2004	19:00:00	2.0	1292.0	112.0	9.4	955.0	103.0	
2	10-03-2004	20:00:00	2.2	1402.0	88.0	9.0	939.0	131.0	
3	10-03-2004	21:00:00	2.2	1376.0	80.0	9.2	948.0	172.0	
4	10-03-2004	22:00:00	1.6	1272.0	51.0	6.5	836.0	131.0	
...	...	...	...	...	...	...	...	...	
9466	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9467	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9468	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9469	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9470	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

9471 rows × 17 columns

In [5]:
```
df.dtypes
```

Out[5]:
```
Date             object
Time             object
CO(GT)          float64
PT08.S1(CO)     float64
NMHC(GT)        float64
C6H6(GT)        float64
PT08.S2(NMHC)   float64
NOx(GT)         float64
PT08.S3(NOx)    float64
NO2(GT)         float64
PT08.S4(NO2)    float64
PT08.S5(O3)     float64
T               float64
RH              float64
AH              float64
Unnamed: 15     float64
Unnamed: 16     float64
dtype: object
```

In [6]:
```
df.info
```

Out[6]:
```
<bound method DataFrame.info of            Date      Time  CO(GT)  PT08.S1(CO)
NMHC(GT)  C6H6(GT)  \
0     10-03-2004  18:00:00     2.6       1360.0     150.0      11.9
1     10-03-2004  19:00:00     2.0       1292.0     112.0       9.4
2     10-03-2004  20:00:00     2.2       1402.0      88.0       9.0
3     10-03-2004  21:00:00     2.2       1376.0      80.0       9.2
4     10-03-2004  22:00:00     1.6       1272.0      51.0       6.5
...          ...       ...     ...          ...       ...       ...
9466         NaN       NaN     NaN          NaN       NaN       NaN
9467         NaN       NaN     NaN          NaN       NaN       NaN
9468         NaN       NaN     NaN          NaN       NaN       NaN
9469         NaN       NaN     NaN          NaN       NaN       NaN
9470         NaN       NaN     NaN          NaN       NaN       NaN

      PT08.S2(NMHC)  NOx(GT)  PT08.S3(NOx)  NO2(GT)  PT08.S4(NO2)  \
0            1046.0    166.0        1056.0    113.0        1692.0
1             955.0    103.0        1174.0     92.0        1559.0
2             939.0    131.0        1140.0    114.0        1555.0
3             948.0    172.0        1092.0    122.0        1584.0
4             836.0    131.0        1205.0    116.0        1490.0
...             ...      ...           ...      ...           ...
9466            NaN      NaN           NaN      NaN           NaN
9467            NaN      NaN           NaN      NaN           NaN
9468            NaN      NaN           NaN      NaN           NaN
9469            NaN      NaN           NaN      NaN           NaN
9470            NaN      NaN           NaN      NaN           NaN

      PT08.S5(O3)     T    RH      AH  Unnamed: 15  Unnamed: 16
0          1268.0  13.6  48.9  0.7578          NaN          NaN
1           972.0  13.3  47.7  0.7255          NaN          NaN
2          1074.0  11.9  54.0  0.7502          NaN          NaN
3          1203.0  11.0  60.0  0.7867          NaN          NaN
4          1110.0  11.2  59.6  0.7888          NaN          NaN
...           ...   ...   ...     ...          ...          ...
9466          NaN   NaN   NaN     NaN          NaN          NaN
9467          NaN   NaN   NaN     NaN          NaN          NaN
9468          NaN   NaN   NaN     NaN          NaN          NaN
9469          NaN   NaN   NaN     NaN          NaN          NaN
9470          NaN   NaN   NaN     NaN          NaN          NaN

[9471 rows x 17 columns]>
```

In [7]:
```
df.head()
```

Out[7]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(
0	10-03-2004	18:00:00	2.6	1360.0	150.0	11.9	1046.0	166.0	1
1	10-03-2004	19:00:00	2.0	1292.0	112.0	9.4	955.0	103.0	1
2	10-03-2004	20:00:00	2.2	1402.0	88.0	9.0	939.0	131.0	1
3	10-03-2004	21:00:00	2.2	1376.0	80.0	9.2	948.0	172.0	1
4	10-03-2004	22:00:00	1.6	1272.0	51.0	6.5	836.0	131.0	1

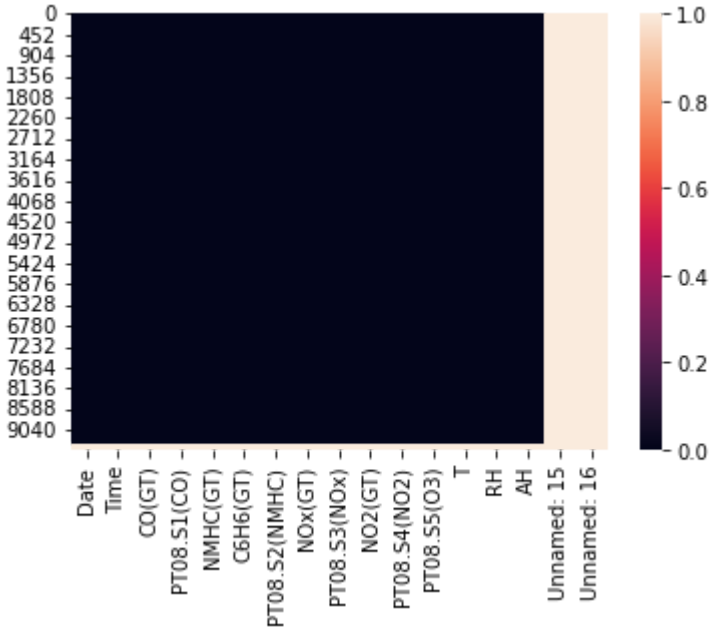
In [9]:
```
df.describe()
```

Out[9]:

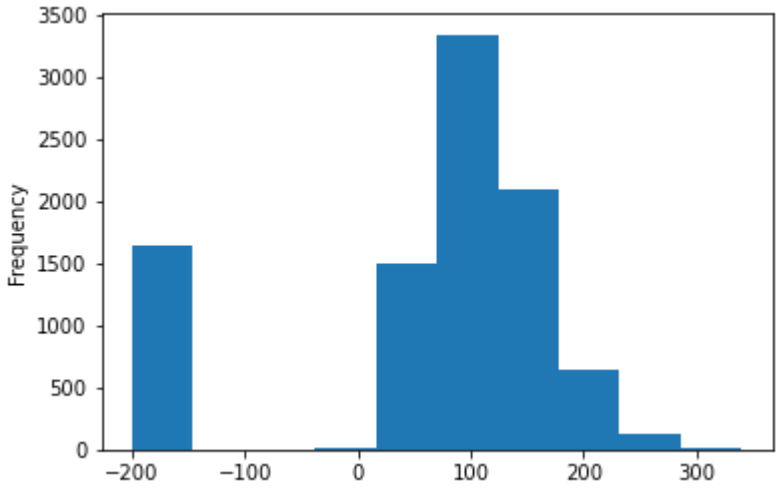
	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(
count	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.000000	9357.0
mean	-34.207524	1048.990061	-159.090093	1.865683	894.595276	168.616971	794.9
std	77.657170	329.832710	139.789093	41.380206	342.333252	257.433866	321.9
min	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.000000	-200.0
25%	0.600000	921.000000	-200.000000	4.000000	711.000000	50.000000	637.0
50%	1.500000	1053.000000	-200.000000	7.900000	895.000000	141.000000	794.0
75%	2.600000	1221.000000	-200.000000	13.600000	1105.000000	284.000000	960.0
max	11.900000	2040.000000	1189.000000	63.700000	2214.000000	1479.000000	2683.0

In [10]:
```
sns.heatmap(df.isnull())
```

Out[10]: <AxesSubplot:>



In [11]:
```
df['NO2(GT)'].plot.hist()
plt.show()
```



In [12]:
```
df['Date'].dtype
```

Out[12]: dtype('O')

In [13]:
```
df['Date'].head()
```

Out[13]:
```
0    10-03-2004
1    10-03-2004
2    10-03-2004
3    10-03-2004
4    10-03-2004
Name: Date, dtype: object
```

In [15]:
```
df['Date'].dropna()
df['Date'] = pd.to_datetime(df['Date'])
```

```
_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
D:\anaconda\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarnin
g: Parsing '30-03-2004' in DD/MM/YYYY format. Provide format or specify infer
_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
D:\anaconda\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarnin
g: Parsing '31-03-2004' in DD/MM/YYYY format. Provide format or specify infer
_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
D:\anaconda\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarnin
g: Parsing '13-04-2004' in DD/MM/YYYY format. Provide format or specify infer
_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
D:\anaconda\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarnin
g: Parsing '14-04-2004' in DD/MM/YYYY format. Provide format or specify infer
_datetime_format=True for consistent parsing.
  cache_array = _maybe_cache(arg, format, cache, convert_listlike)
D:\anaconda\lib\site-packages\pandas\core\tools\datetimes.py:1047: UserWarnin
g: Parsing '15-04-2004' in DD/MM/YYYY format. Provide format or specify infer
```

In [16]:
```
df['Date'].head()
```

Out[16]:
```
0   2004-10-03
1   2004-10-03
2   2004-10-03
3   2004-10-03
4   2004-10-03
Name: Date, dtype: datetime64[ns]
```

In [17]:
```
df.head()
```

Out[17]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3
0	2004-10-03	18:00:00	2.6	1360.0	150.0	11.9	1046.0	166.0	1
1	2004-10-03	19:00:00	2.0	1292.0	112.0	9.4	955.0	103.0	1
2	2004-10-03	20:00:00	2.2	1402.0	88.0	9.0	939.0	131.0	1
3	2004-10-03	21:00:00	2.2	1376.0	80.0	9.2	948.0	172.0	1
4	2004-10-03	22:00:00	1.6	1272.0	51.0	6.5	836.0	131.0	1

In [18]:
```
df['Time'].dtype
```

Out[18]:
```
dtype('O')
```

In [19]:
```
df['Time'].head()
```

Out[19]:
```
0    18:00:00
1    19:00:00
2    20:00:00
3    21:00:00
4    22:00:00
Name: Time, dtype: object
```

In [20]:
```
df['Time'].dropna()
```

Out[20]:
```
0       18:00:00
1       19:00:00
2       20:00:00
3       21:00:00
4       22:00:00
          ...
9352    10:00:00
9353    11:00:00
9354    12:00:00
9355    13:00:00
9356    14:00:00
Name: Time, Length: 9357, dtype: object
```

In [21]:
```
df['Time'].dropna()
df['Time'] = pd.to_datetime(df['Time'])
```

In [22]:
```
df['day']=df["Date"].apply(lambda x: x.day)
df['month']=df["Date"].apply(lambda x: x.month)
df['year']=df["Date"].apply(lambda x: x.year)
```

In [23]:
```
df['Time'].dropna()
df['Time'] = pd.to_datetime(df['Time'])
```

In [24]:
```
df['hour']=df["Date"].apply(lambda x: x.hour)
df['minute']=df["Date"].apply(lambda x: x.minute)
df['second']=df["Date"].apply(lambda x: x.second)
```

In [25]:
```
df.head()
```

Out[25]:

NOx)	NO2(GT)	...	RH	AH	Unnamed: 15	Unnamed: 16	day	month	year	hour	minute	second
056.0	113.0	...	48.9	0.7578	NaN	NaN	3.0	10.0	2004.0	0.0	0.0	0.0
174.0	92.0	...	47.7	0.7255	NaN	NaN	3.0	10.0	2004.0	0.0	0.0	0.0
140.0	114.0	...	54.0	0.7502	NaN	NaN	3.0	10.0	2004.0	0.0	0.0	0.0
092.0	122.0	...	60.0	0.7867	NaN	NaN	3.0	10.0	2004.0	0.0	0.0	0.0
205.0	116.0	...	59.6	0.7888	NaN	NaN	3.0	10.0	2004.0	0.0	0.0	0.0

In [26]:
```
df['Time'].dropna()
df['Time'] = pd.to_datetime(df['Time'])
```

In [27]:
```
df['hour']=df["Time"].apply(lambda x: x.hour)
df['minute']=df["Time"].apply(lambda x: x.minute)
df['second']=df["Time"].apply(lambda x: x.second)
```

In [28]:
```
df.head()
```

Out[28]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3
0	2004-10-03	2022-06-17 18:00:00	2.6	1360.0	150.0	11.9	1046.0	166.0	1
1	2004-10-03	2022-06-17 19:00:00	2.0	1292.0	112.0	9.4	955.0	103.0	1
2	2004-10-03	2022-06-17 20:00:00	2.2	1402.0	88.0	9.0	939.0	131.0	1
3	2004-10-03	2022-06-17 21:00:00	2.2	1376.0	80.0	9.2	948.0	172.0	1
4	2004-10-03	2022-06-17 22:00:00	1.6	1272.0	51.0	6.5	836.0	131.0	1

5 rows × 23 columns

In [29]:
```
df = df.drop(['Date', 'Time'],axis=1)
```

In [36]:
```
print("Number of null values in each column:\n{}".format(df.isnull().sum()))
```

```
Number of null values in each column:
CO(GT)              0
PT08.S1(CO)         0
NMHC(GT)            0
C6H6(GT)            0
PT08.S2(NMHC)       0
NOx(GT)             0
PT08.S3(NOx)        0
NO2(GT)             0
PT08.S4(NO2)        0
PT08.S5(O3)         0
T                   0
RH                  0
AH                  0
Unnamed: 15      9471
Unnamed: 16      9471
day                 0
month               0
year                0
hour                0
minute              0
second              0
dtype: int64
```

In [33]:
```
for i in df.columns:
    df[i]=df[i].fillna(df[i].mean())
```

In [34]:
```
feat=df.drop(['AH','Unnamed: 15','Unnamed: 16'],axis=1)
val=df['AH'].values
```

In [35]:
```
feat.head()
```

Out[35]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)
0	2.6	1360.0	150.0	11.9	1046.0	166.0	1056.0	113.0
1	2.0	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0
2	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0
3	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0
4	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0

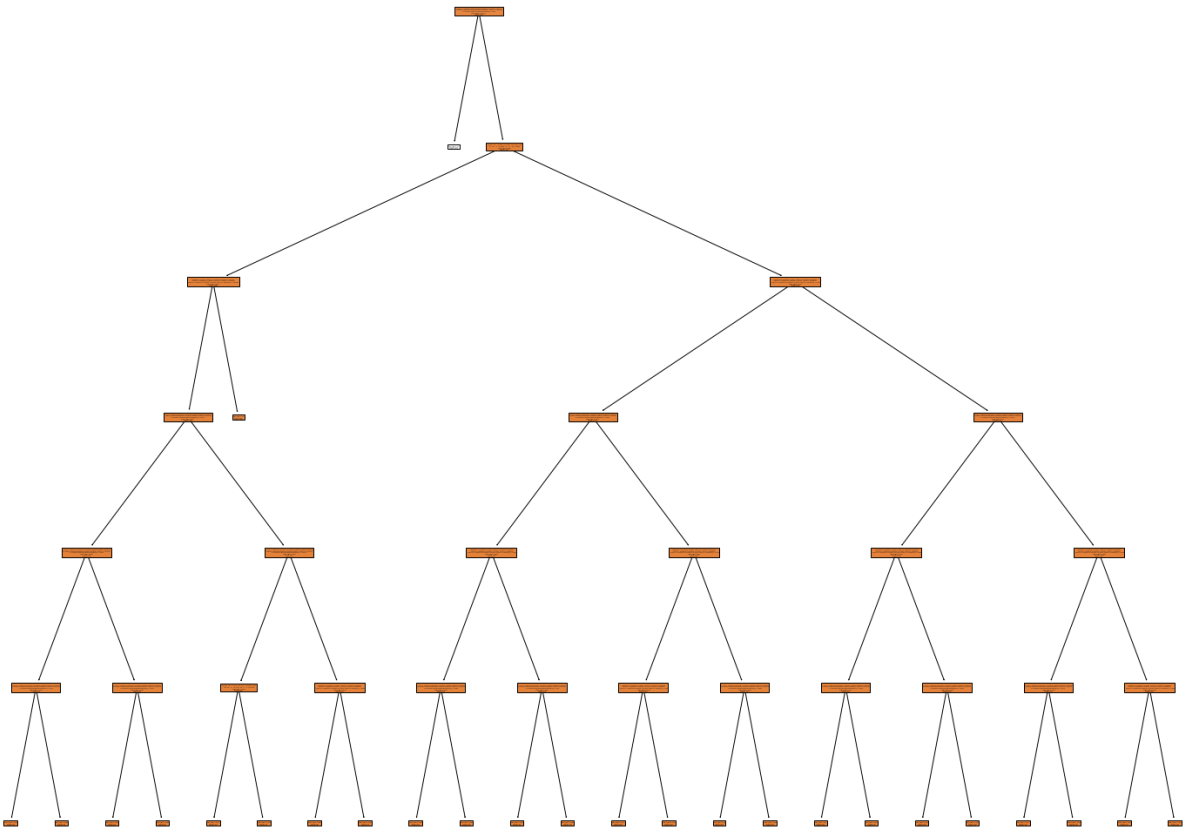
In [41]:
```
(train_feat,test_feat,train_classes,test_classes) = train_test_split(feat,val,rar
m = DecisionTreeRegressor(max_depth=6).fit(train_feat,train_classes)
```

In [42]:
```
ypred=m.predict(test_feat)
```

In [43]:
```
acc = mean_squared_error(test_classes,ypred)
acc
```

Out[43]:
```
0.027866139716966495
```

In [51]:
```
from sklearn import tree
fig = plt.figure(figsize=(25,20))
_= tree.plot_tree(m,
                  feature_names=feat.values,
                  #class_names=val,
                  filled=True)
```



In [53]:
```
text_representation = tree.export_text(m)
print(text_representation)
```

```
|--- feature_4 <= 91.50
|   |--- value: [-200.00]
|--- feature_4 >  91.50
|   |--- feature_3 <= 1.88
|   |   |--- feature_9 <= 914.04
|   |   |   |--- feature_8 <= 967.00
|   |   |   |   |--- feature_8 <= 849.50
|   |   |   |   |   |--- feature_8 <= 729.00
|   |   |   |   |   |   |--- value: [0.30]
|   |   |   |   |   |--- feature_8 >  729.00
|   |   |   |   |   |   |--- value: [0.43]
|   |   |   |   |--- feature_8 >  849.50
|   |   |   |   |   |--- feature_8 <= 879.50
|   |   |   |   |   |   |--- value: [0.55]
|   |   |   |   |   |--- feature_8 >  879.50
|   |   |   |   |   |   |--- value: [0.65]
|   |   |   |--- feature_8 >  967.00
|   |   |   |   |--- feature_8 <= 1294.50
|   |   |   |   |   |--- feature_2 <= -96.50
|   |   |   |   |   |   |--- value: [0.92]
|   |   |   |   |   |--- feature_2 >  -96.50
|   |   |   |   |   |   |--- value: [0.73]
|   |   |   |   |--- feature_8 >  1294.50
|   |   |   |   |   |--- feature_10 <= 23.95
|   |   |   |   |   |   |--- value: [1.25]
|   |   |   |   |   |--- feature_10 >  23.95
|   |   |   |   |   |   |--- value: [1.83]
|   |   |--- feature_9 >  914.04
|   |   |   |--- value: [-6.84]
|   |--- feature_3 >  1.88
|   |   |--- feature_10 <= 16.75
|   |   |   |--- feature_11 <= 50.85
|   |   |   |   |--- feature_10 <= 13.75
|   |   |   |   |   |--- feature_11 <= 40.95
|   |   |   |   |   |   |--- value: [0.39]
|   |   |   |   |   |--- feature_11 >  40.95
|   |   |   |   |   |   |--- value: [0.50]
|   |   |   |   |--- feature_10 >  13.75
|   |   |   |   |   |--- feature_11 <= 35.50
|   |   |   |   |   |   |--- value: [0.46]
|   |   |   |   |   |--- feature_11 >  35.50
|   |   |   |   |   |   |--- value: [0.77]
|   |   |   |--- feature_11 >  50.85
|   |   |   |   |--- feature_10 <= 11.15
|   |   |   |   |   |--- feature_10 <= 6.35
|   |   |   |   |   |   |--- value: [0.57]
|   |   |   |   |   |--- feature_10 >  6.35
|   |   |   |   |   |   |--- value: [0.78]
|   |   |   |   |--- feature_10 >  11.15
|   |   |   |   |   |--- feature_11 <= 68.05
|   |   |   |   |   |   |--- value: [0.96]
|   |   |   |   |   |--- feature_11 >  68.05
|   |   |   |   |   |   |--- value: [1.17]
```

```
|   |   |--- feature_10 >  16.75
|   |   |   |--- feature_11 <= 47.25
|   |   |   |   |--- feature_10 <= 23.15
|   |   |   |   |   |--- feature_11 <= 36.25
|   |   |   |   |   |   |--- value: [0.70]
|   |   |   |   |   |--- feature_11 >  36.25
|   |   |   |   |   |   |--- value: [1.00]
|   |   |   |   |--- feature_10 >  23.15
|   |   |   |   |   |--- feature_11 <= 30.25
|   |   |   |   |   |   |--- value: [1.09]
|   |   |   |   |   |--- feature_11 >  30.25
|   |   |   |   |   |   |--- value: [1.39]
|   |   |   |--- feature_11 >  47.25
|   |   |   |   |--- feature_10 <= 21.35
|   |   |   |   |   |--- feature_11 <= 61.15
|   |   |   |   |   |   |--- value: [1.18]
|   |   |   |   |   |--- feature_11 >  61.15
|   |   |   |   |   |   |--- value: [1.53]
|   |   |   |   |--- feature_10 >  21.35
|   |   |   |   |   |--- feature_10 <= 25.45
|   |   |   |   |   |   |--- value: [1.59]
|   |   |   |   |   |--- feature_10 >  25.45
|   |   |   |   |   |   |--- value: [1.85]
```

In [ ]: