```
import pandas as pd
import numpy as np
import random

df = pd.DataFrame(
                  [
                      [6.82, 118, 0],
                      [6.36, 125, 1],
                      [5.39, 99, 1],
                      [5.50, 106, 1],
                      [6.39, 148, 0],
                      [9.13, 148, 1],
                      [7.17, 147, 1],
                      [7.72, 72, 0]
                   ], columns = ['cgpa',  'iq', 'is_placed']
)

df
```

	cgpa	iq	is_placed
0	6.82	118	0
1	6.36	125	1
2	5.39	99	1
3	5.50	106	1
4	6.39	148	0
5	9.13	148	1
6	7.17	147	1
7	7.72	72	0

```
# initial prediction

df['pre1(log-odds)'] = np.log(5/3)
df
```

	cgpa	iq	is_placed	pre1(log-odds)
0	6.82	118	0	0.510826
1	6.36	125	1	0.510826
2	5.39	99	1	0.510826
3	5.50	106	1	0.510826
4	6.39	148	0	0.510826
5	9.13	148	1	0.510826
6	7.17	147	1	0.510826
7	7.72	72	0	0.510826

```
# convert log odds to probability
df['pre1(probability)'] = 1/(1+np.exp(-np.log(5/3)))
```

```
df['pre1(probability)'] = 1/(1+np.exp(-np.log(5/3)))
df
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)
0	6.82	118	0	0.510826	0.625
1	6.36	125	1	0.510826	0.625
2	5.39	99	1	0.510826	0.625
3	5.50	106	1	0.510826	0.625
4	6.39	148	0	0.510826	0.625
5	9.13	148	1	0.510826	0.625
6	7.17	147	1	0.510826	0.625
7	7.72	72	0	0.510826	0.625

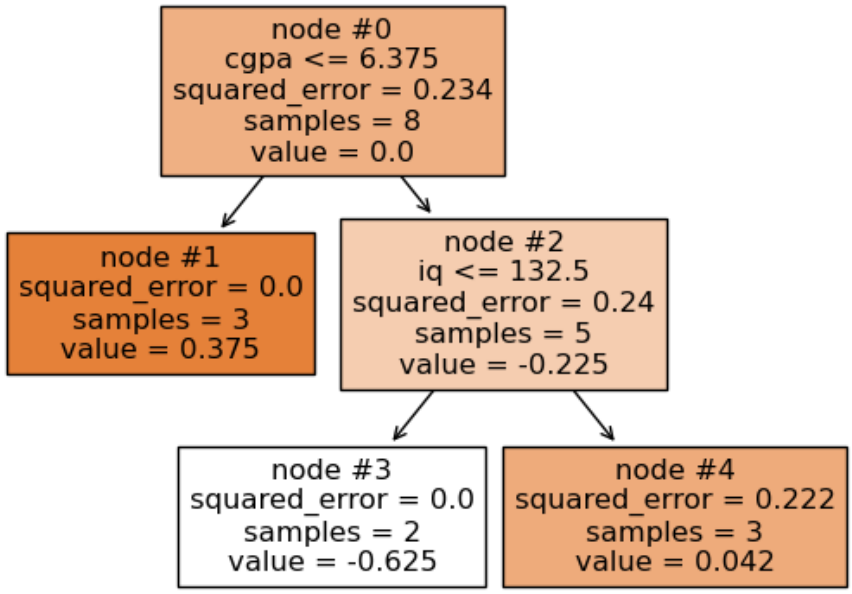
```
# calculating residual for stage 1
df['res1'] = df['is_placed'] - df['pre1(probability)']
df
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1
0	6.82	118	0	0.510826	0.625	-0.625
1	6.36	125	1	0.510826	0.625	0.375
2	5.39	99	1	0.510826	0.625	0.375
3	5.50	106	1	0.510826	0.625	0.375
4	6.39	148	0	0.510826	0.625	-0.625
5	9.13	148	1	0.510826	0.625	0.375
6	7.17	147	1	0.510826	0.625	0.375
7	7.72	72	0	0.510826	0.625	-0.625

```
# training the first decision tree
from sklearn.tree import DecisionTreeRegressor

reg1 = DecisionTreeRegressor(max_leaf_nodes=3, random_state=1)

reg1.fit(df.iloc[:,0:2].values, df.iloc[:,-1].values)
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(max_leaf_nodes=3, random_state=1)
```

```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plot_tree(reg1, feature_names=['cgpa','iq'],filled=True, node_ids=True)
plt.show()
```




```
df['leaf_entry1'] = reg1.apply(df.iloc[:,0:2])
df
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has feature names,
  warnings.warn(
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1	leaf_entry1
0	6.82	118	0	0.510826	0.625	-0.625	3
1	6.36	125	1	0.510826	0.625	0.375	1
2	5.39	99	1	0.510826	0.625	0.375	1
3	5.50	106	1	0.510826	0.625	0.375	1
4	6.39	148	0	0.510826	0.625	-0.625	4
5	9.13	148	1	0.510826	0.625	0.375	4
6	7.17	147	1	0.510826	0.625	0.375	4
7	7.72	72	0	0.510826	0.625	-0.625	3

```
def return_logs(leaf):
  temp_df = df[df['leaf_entry1'] == leaf]
  num = temp_df['res1'].sum()

  den = sum(temp_df['pre1(probability)'] * (1 - temp_df['pre1(probability)']))
  return round(num/den,2)
```

```
df['pre2(log-odds)'] = df['pre1(log-odds)'] + df['leaf_entry1'].apply(return_logs)
```

```
0    -2.67
1     1.60
2     1.60
3     1.60
4     0.18
```

```
5    0.18
6    0.18
7   -2.67
Name: leaf_entry1, dtype: float64
```

```
df['pre2(probability)'] = 1/(1+np.exp(-df['pre2(log-odds)']))
df
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1	leaf_entry1	pre2(log odds
0	6.82	118	0	0.510826	0.625	-0.625	3	-2.1591
1	6.36	125	1	0.510826	0.625	0.375	1	2.1108
2	5.39	99	1	0.510826	0.625	0.375	1	2.1108
3	5.50	106	1	0.510826	0.625	0.375	1	2.1108
4	6.39	148	0	0.510826	0.625	-0.625	4	0.6908
5	9.13	148	1	0.510826	0.625	0.375	4	0.6908
6	7.17	147	1	0.510826	0.625	0.375	4	0.6908
7	7.72	72	0	0.510826	0.625	-0.625	3	-2.1591

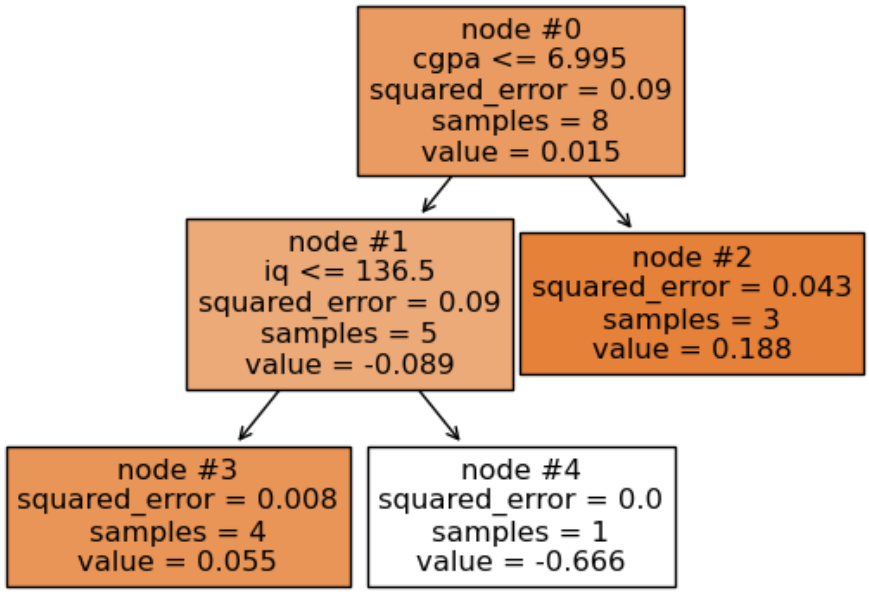
```
df['res2'] = df['is_placed'] - df['pre2(probability)']
df
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1	leaf_entry1	pre2(log odds
0	6.82	118	0	0.510826	0.625	-0.625	3	-2.1591
1	6.36	125	1	0.510826	0.625	0.375	1	2.1108
2	5.39	99	1	0.510826	0.625	0.375	1	2.1108
3	5.50	106	1	0.510826	0.625	0.375	1	2.1108
4	6.39	148	0	0.510826	0.625	-0.625	4	0.6908
5	9.13	148	1	0.510826	0.625	0.375	4	0.6908
6	7.17	147	1	0.510826	0.625	0.375	4	0.6908
7	7.72	72	0	0.510826	0.625	-0.625	3	-2.1591

```
reg2 = DecisionTreeRegressor(max_leaf_nodes=3, random_state=1)

reg2.fit(df.iloc[:,0:2].values, df.iloc[:,-1].values)
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(max_leaf_nodes=3, random_state=1)
```

```
plot_tree(reg2, feature_names=['cgpa','iq'],filled=True, node_ids=True)
plt.show()
```




```
df['leaf_entry2'] = reg2.apply(df.iloc[:,0:2])
df
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has
  warnings.warn(
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1	leaf_entry1	pre2(log odds
0	6.82	118	0	0.510826	0.625	-0.625	3	-2.1591
1	6.36	125	1	0.510826	0.625	0.375	1	2.1108
2	5.39	99	1	0.510826	0.625	0.375	1	2.1108
3	5.50	106	1	0.510826	0.625	0.375	1	2.1108
4	6.39	148	0	0.510826	0.625	-0.625	4	0.6908
5	9.13	148	1	0.510826	0.625	0.375	4	0.6908
6	7.17	147	1	0.510826	0.625	0.375	4	0.6908
7	7.72	72	0	0.510826	0.625	-0.625	3	-2.1591

```
def return_logs(leaf):
  num = df[df['leaf_entry2'] == leaf]['res2'].sum()
  den = sum(df[df['leaf_entry2'] == leaf]['pre2(probability)'] * (1 - df[df['leaf_entry2'] == leaf
  return round(num/den,2)
```

```
df['pre3(log-odds)'] = df['pre1(log-odds)'] + df['pre2(log-odds)'] + df['leaf_entry2'].apply(retur
```

```
df['pre3(probability)'] = 1/(1+np.exp(-df['pre3(log-odds)']))
df
```

	cgpa	iq	is_placed	pre1(log-odds)	pre1(probability)	res1	leaf_entry1	pre2(log-odds)	pı
0	6.82	118	0	0.510826	0.625	-0.625	3	-2.159174	
1	6.36	125	1	0.510826	0.625	0.375	1	2.110826	
2	5.39	99	1	0.510826	0.625	0.375	1	2.110826	
3	5.50	106	1	0.510826	0.625	0.375	1	2.110826	
4	6.39	148	0	0.510826	0.625	-0.625	4	0.690826	
5	9.13	148	1	0.510826	0.625	0.375	4	0.690826	
6	7.17	147	1	0.510826	0.625	0.375	4	0.690826	
7	7.72	72	0	0.510826	0.625	-0.625	3	-2.159174	

```
df['res_final'] = df['is_placed'] - df['pre3(probability)']

df[['res1','res2','res_final']]
```

	res1	res2	res_final
0	-0.625	-0.103477	-0.255717
1	0.375	0.108049	0.039104
2	0.375	0.108049	0.039104
3	0.375	0.108049	0.039104
4	-0.625	-0.666151	-0.142052
5	0.375	0.333849	0.095207
6	0.375	0.333849	0.095207
7	-0.625	-0.103477	-0.354722

Start coding or generate with AI.