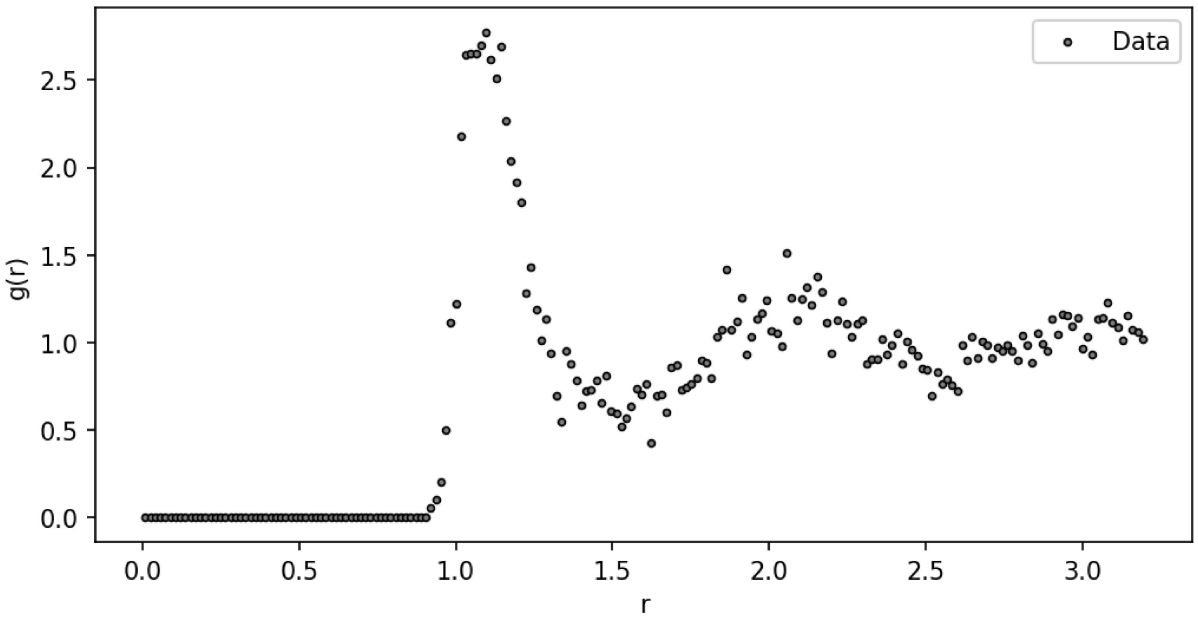
M5-L2 Problem 1

256 particles of liquid argon are simulated at 100K. A radial distribution function $g(r)$ describes the density of particles a distance of r from each particle in the system. When an $g(r)$ is computed in a simulation, it is done by creating a histogram of particle distances for a single simulation frame, resulting in a noisy function that is most often averaged over several frames.

Given g(r) vs. r data for a single frame, you will train a decision tree regressor to represent the underlying function.

First, run the cell below to load the data, etc.:

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor,plot_tree

r = np.array([0.008,0.024,0.04,0.056,0.072,0.088,0.104,0.12,0.136,0.152,0.168,0.184
g = np.array([0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,

def plot(r, g, dt = None):
    if dt is not None:
        plt.figure(figsize=(12,3),dpi=150)
        plt.subplot(121)
        rs = np.linspace(0,4,1000)
        gs = dt.predict(rs.reshape(-1,1))
        plt.plot(rs,gs,color="red",label="Regression Tree",alpha=0.7)
    else:
        plt.figure(figsize=(8,4),dpi=150)

    plt.scatter(r,g,s=8,c="gray", label="Data", edgecolors="black",linewidths=.8)
    plt.legend(loc="upper right")
    plt.xlabel("r")
    plt.ylabel("g(r)")

    if dt is not None:
        plt.subplot(122)
        plot_tree(dt)
        plt.title(f"Tree max. depth: {dt.max_depth}",y=-.2)

    plt.show()

plot(r,g)
```



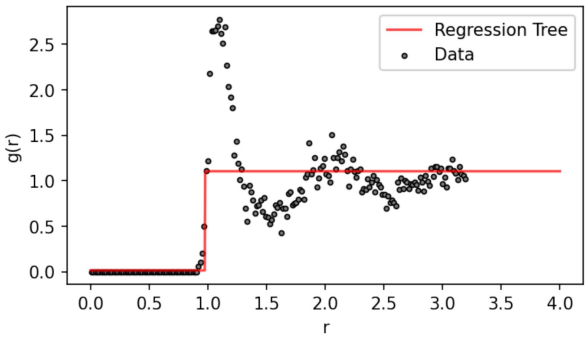
Training regression trees

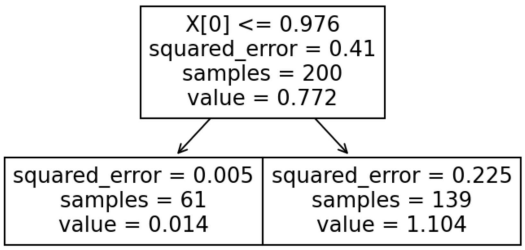
For input `r` and output `g`, train a `DecisionTreeRegressor()` to perform the regression with `max_depth` values of 1, 2, 6, 10.

Complete the code below, which will plot your decision tree results and visualize the tree. Name each decision tree within the loop `dt`.

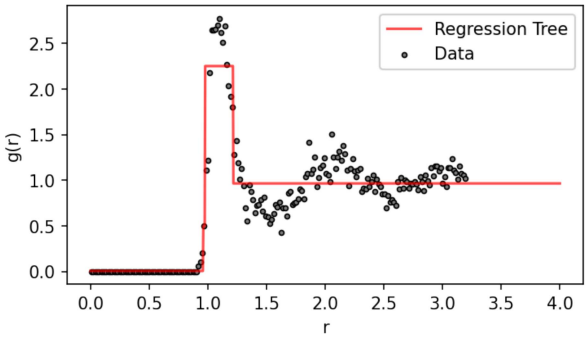
Note: you may need to resize the input `r` as `r.reshape(-1,1)` before passing it as input into the fitting function.

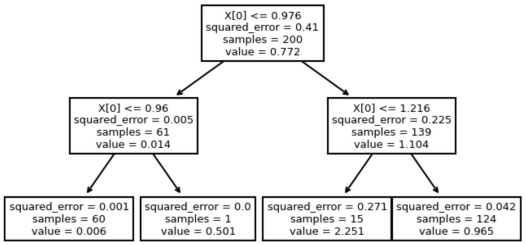
In [4]:
```
for max_depth in [1, 2, 6, 10]:
    # YOUR CODE GOES HERE
    # Create and fit `dt`
    dt = DecisionTreeRegressor(max_depth=max_depth)
    r = r.reshape(-1,1)
    dt.fit(r,g)
    dt.predict(r)

    plot(r,g,dt)
```



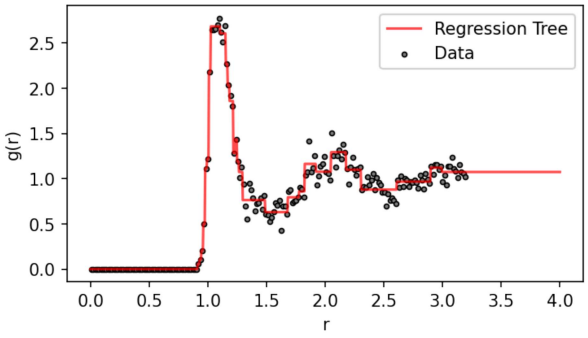


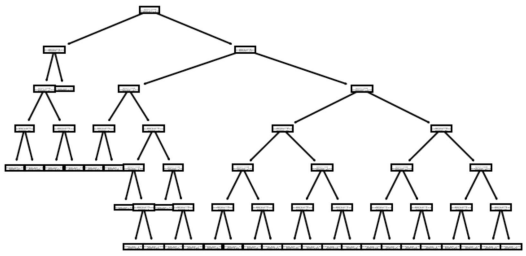
Tree max. depth: 1



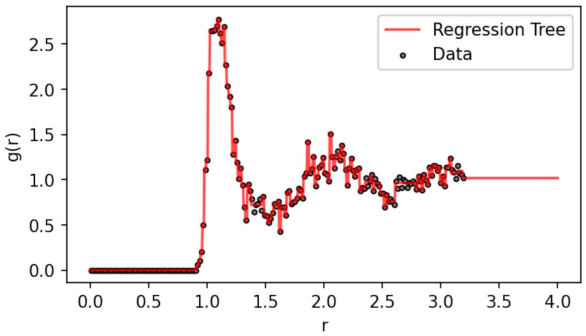


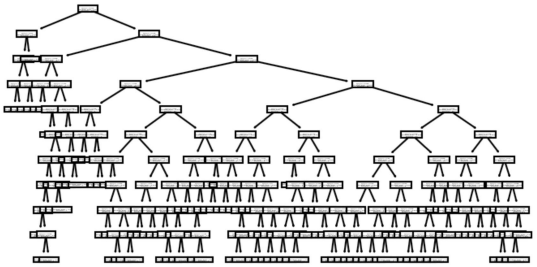
Tree max. depth: 2





Tree max. depth: 6





Tree max. depth: 10