In [2]:
```
import os

os.makedirs(os.path.join('..','data'), exist_ok=True)
data_file = os.path.join('..','data','house_tiny.csv')
with open(data_file, 'w') as f:
    f.write('''NumRooms,RoofType,Price
NA,NA,127500
2,NA,106000
4,Slate,178100
NA,NA,140000''')
```

In [4]:
```
import pandas as pd

data = pd.read_csv(data_file)
print(data)
```

```
   NumRooms RoofType   Price
0       NaN      NaN  127500
1       2.0      NaN  106000
2       4.0    Slate  178100
3       NaN      NaN  140000
```

In [5]:
```
inputs, targets = data.iloc[:,0:2], data.iloc[:,2]
inputs = pd.get_dummies(inputs, dummy_na=True)
print(inputs)
```

```
   NumRooms  RoofType_Slate  RoofType_nan
0       NaN           False          True
1       2.0           False          True
2       4.0            True         False
3       NaN           False          True
```

In [6]:
```
inputs = inputs.fillna(inputs.mean())
print(inputs)
```

```
   NumRooms  RoofType_Slate  RoofType_nan
0       3.0           False          True
1       2.0           False          True
2       4.0            True         False
3       3.0           False          True
```

In [7]:
```
import torch

X = torch.tensor(inputs.to_numpy(dtype=float))
y = torch.tensor(targets.to_numpy(dtype=float))
X,y
```

Out[7]:
```
(tensor([[3., 0., 1.],
         [2., 0., 1.],
         [4., 1., 0.],
         [3., 0., 1.]], dtype=torch.float64),
 tensor([127500., 106000., 178100., 140000.], dtype=torch.float64))
```

Discussion: First, I created a data file, and I was surprised to find that, despite using what I felt was a somewhat forced method, the CSV file came out correctly when printed. (I used a multi-line string for the input and separated the fields with newlines, which is why I described it as a forced method.) Then, I split the 'Rooftype' column into 'Slate' and 'NaN', dividing the table into True/False values and converted it into a tensor. Through this process, I realized that human-friendly data and machine-friendly data have different goals.