LogNormalDataset-modular

December 6, 2021

0.0.1 A First, Naive Learned Index on Log normal dataset

This is an implementation of learning indexes using neural networks as described in the recent paper from google.

some variable initialisations

[ ]:
```
mu, sigma = 3., 1. # mean and standard deviation
num_datapoints = 10000
```

Importing libraries and preparing training data to be indexed

[ ]:
```
%matplotlib inline
%load_ext autoreload
%autoreload 2

import numpy as np
s = np.random.lognormal(mu, sigma, num_datapoints)
np_data = np.asarray(sorted(s))
np_data.shape
```

[ ]:
```
(10000,)
```

Using pytorch to train a neural network to learn the indexes of the dataset (s)

[ ]:
```
import torch
from index_learner import *
D_in, H, D_out =  1, 100, 1
model = torch.nn.Sequential(
        torch.nn.Linear(D_in, H),
        torch.nn.ReLU(),
        torch.nn.Linear(H, D_out),
    )
x = torch.FloatTensor(np_data.reshape(num_datapoints,1)[:,:])
plot_step,plot_lossess,model = learn_index(num_datapoints,x,model)
```

```
0 33425470.0
1000 4275164.5
2000 235495.9375
```

```
3000 22872.95703125
4000 4333.9638671875
```

Time taken by model to predict index positions for all points in the dataset (s)

[ ]:
```
%%time
predicted_index,error_predicted_index = predict_indexes(num_datapoints,model,x)
```

```
Total datapoint: 10000
Wall time: 545 ms
```

Various plots to visually understand the dataset, model trainig and index predictions. Plot 2 shows that the error in predicted_index is very low usually around zero for most of the dataset. This is an encouraging result for a naive approach in using neural network for learning indexes.

[ ]:
```
plot_results(num_datapoints,predicted_index,error_predicted_index
                ,plot_step,plot_lossess,np_data)
```