SYDE Coding Project 2

Description

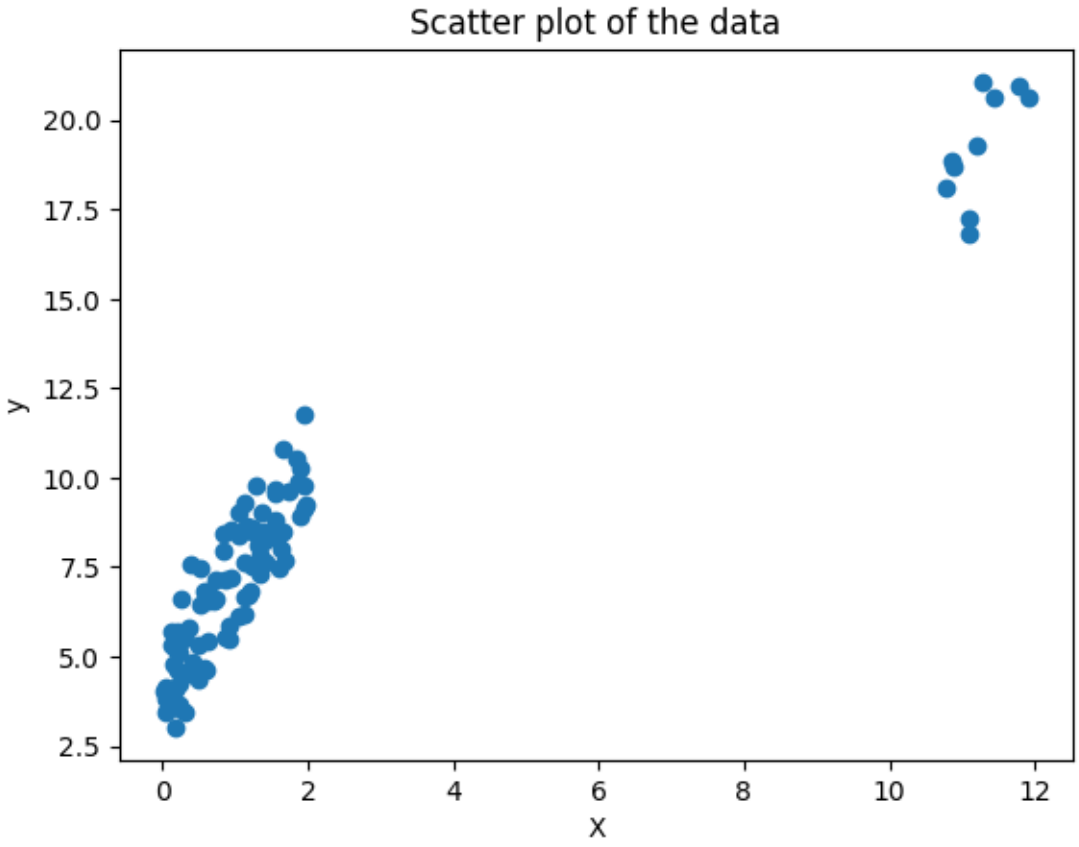
The aim of this project is to help you understand the limitation of model representation capability and the importance of initialization.

Format: a Jupyter notebook containing code. Make sure you code can be run in colab.
Due: Jan 31th 11:59pm

Steps

1. Modify your code from Project 1 for this project. We will turn our regression model into a simple generative model using MLP.

2. Change the sample data generation in your code, so it now produces sample points from two different regression lines (I am using 90 samples for the first line and 10 points for the line). Please make sure that the input from the two regressions do not overlap. It should look similar to the following figure, and you don't have to replicate the exact same samples in my figure:



3. Run your model from project 1 on this data, report a graph and print the loss.

4. Make your model into a 3 layer MLP with "relu" activation in between. The 3 layer design is to simulate an encoder-decoder style architecture. Use two hyper-parameters to control the number of neurons in the encoder and decoder. See sample code below:

```
self.fc1 = nn.Linear(1, hidden_enc)
self.fc2 = nn.Linear(hidden_enc, hidden_dec)
self.fc3 = nn.Linear(hidden_dec, 1)
```

5. The number of hidden neurons defines the representation capability of your model. Experiment your model with the following hyper-parameter values to see how encoder and decoder width affects your results. For each model, print the loss and display the generated points (use plt.scatter for the predicted points).

hidden_enc	hidden_dec
1	1
1	2
2	1
2	2
10	10

6. Repeat step 5 until you see how random initialization affects the result. Display ONE figure that shows an inferior model due to bad initialization (not due to representation limitation).