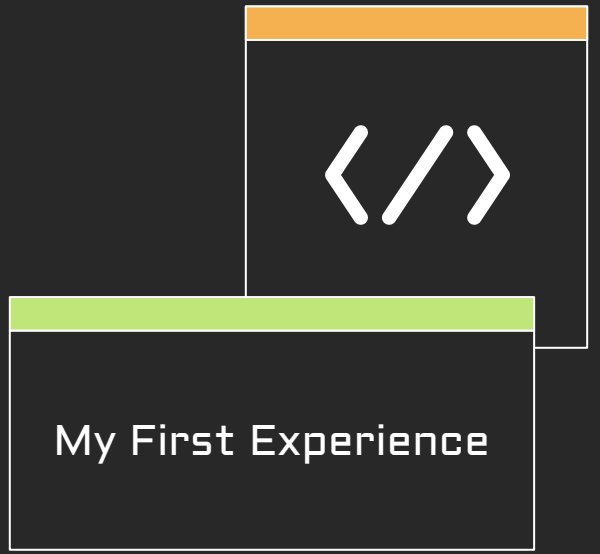
1 0 0 1 0 1 0 1 0 1 0 1 0 1 0 0

Pytorch



1 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0

Installing Pytorch on Mac

- Mac is often known to be notoriously difficult when it comes to installing well known programs in Comp Sci
- Going to the pytorch website showed a lot of prerequisites before downloading the main program
  - Installing on a virtual machine
    - Anaconda
    - PyPi
  - What do I chose?

1 0 0 1 0 1 0 1 0 1 0 1 0 0

1 0 1 0 0 1 0 1 0 1 0 0 1 0 1 0

Virtual Machine Exposure

- First time setting up a virtual machine
  - Google TechX
    - Google cloud console
- I had only been recently exposed to VMs the last couple of months
- From what I learned a VM is:
  - An isolated computing environment created by abstracting resources from a physical machine
- Most of the resources I looked into was linking a VM to a Jupyter Notebook via Anaconda
  - How am I going to achieve this?

YouTube!

- Google is a great way to look for answers and resources
- I found a video to set up both a VM and Pytorch on Jupyter Notebook
- The video was a awesome set by set tutorial that helped me visualize the set up
- There was an included Github link that further explained what each line of code did along with a test example

https://github.com/mrdbourke/pytorch-apple-silicon

What did the Process Look Like?

1 0 1 0 1 0 1 1 1 0 0 0 1 1 1 0

[1]:
```
import torch
import numpy as np
import pandas as pd
import sklearn
import matplotlib.pyplot as plt

print(f"PyTorch version: {torch.__version__}")

# Check PyTorch has access to MPS (Metal Performance Shader, Apple's GPU architecture)
print(f"Is MPS (Metal Performance Shader) built? {torch.backends.mps.is_built()}")
print(f"Is MPS available? {torch.backends.mps.is_available()}")

# Set the device
device = "mps" if torch.backends.mps.is_available() else "cpu"
print(f"Using device: {device}")
```

```
PyTorch version: 2.2.1
Is MPS (Metal Performance Shader) built? True
Is MPS available? True
Using device: mps
```

[2]:
```
import torch

# Set the device
device = "mps" if torch.backends.mps.is_available() else "cpu"

# Create data and send it to the device
x = torch.rand(size=(3, 4)).to(device)
x
```

[2]:
```
tensor([[0.4009, 0.1407, 0.8364, 0.0629],
        [0.1625, 0.8269, 0.7616, 0.2799],
        [0.6965, 0.1126, 0.6350, 0.1853]], device='mps:0')
```

[3]:
```
pip install torch_geometric
```

- After following the tutorial I needed to test that pytorch was working
  - Spoiler: It Did!
- I was excited to learn more and followed along with the rest of the examples and tutorials on Pytorch

1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 0

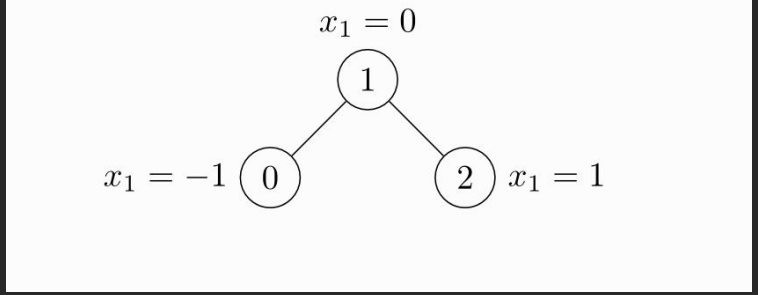
Introduction by Examples

- I proceeded to go down the list on Pytorch
  - Data Handling of Graphs
  - Common Benchmark Datasets
  - Mini-batches
  - Data Transforms
  - Learning Methods on Graphs
- Each had great examples and code follow alongs

```python
import torch
from torch_geometric.data import Data

edge_index = torch.tensor([[0, 1],
                           [1, 0],
                           [1, 2],
                           [2, 1]], dtype=torch.long)
x = torch.tensor([[-1], [0], [1]], dtype=torch.float)

data = Data(x=x, edge_index=edge_index.t().contiguous())
>>> Data(edge_index=[2, 4], x=[3, 1])
```



Where am I now?

Currently:

- Going back and forth between Colab and Jupyter Notebooks
- Using extra resources such as the CS224W Machine Learning with Graphs slides
- GitHub examples
  - https://github.com/pyg-team/pytorch_geometric/tree/master/examples