Multinomial logistic regression

```python
import numpy as np
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import matplotlib.pyplot as plt
```

```python
x_data = [[1, 2, 1, 1],
          [2, 1, 3, 2],
          [3, 1, 3, 4],
          [4, 1, 5, 5],
          [1, 7, 5, 5],
          [1, 2, 5, 6],
          [1, 6, 6, 6],
          [1, 7, 7, 7]]
y_data = [2, 2, 2, 1, 1, 1, 0, 0]
```

```python
x_train = torch.FloatTensor(x_data)
y_train = torch.LongTensor(y_data)
```

```python
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
x_pca = pca.fit_transform(x_train)

for label in range(3):
    x_one_pre = x_pca[y_train==label]
    plt.scatter(x_one_pre[:, 0], x_one_pre[:, 1])
```

cost

```python
from IPython.display import Image
image_url = 'https://bitbucket.org/hyuk125/lg_dic/raw/99785e9d01523e8bb6bf78d1
Image(image_url)
```

Pytorch has cross entropy function:
torch.nn.functional.cross_entropy(input, target, ...)

```python
from IPython.display import Image
image_url = 'https://bitbucket.org/hyuk125/lg_dic/raw/99785e9d01523e8bb6bf78d1
Image(image_url)
```

```
predict = torch.FloatTensor(np.array([[0.5, 0.2], [0.1, 0.7], [0.9, 2.3]]))
label = torch.FloatTensor(np.array([0, 0, 1])).long()
```

```
 F.cross_entropy(predict, label)
```

과제2. 위 데이터를 nn.Module을 활용하여 multinomial logisitc regression하는 코드를 작성하시오

단, epoch=1000, learning rate=1로 하시오

```
class my_multilogic(nn.Module):








```

```
model = my_multilogic()
```

```
# Training











```

결과 출력

```
pca = PCA(n_components=2)
x_pca = pca.fit_transform(x_train)
```

```
torch.max(model(x_train), axis=1).indices == 2
```

```
for predict in range(3):
    x_one_pre = x_pca[torch.max(model(x_train), axis=1).indices==predict]
    plt.scatter(x_one_pre[:, 0], x_one_pre[:, 1])
```

Accuracy

```
hypothesis = model(x_train)

prediction = torch.max(model(x_train), axis=1).indices

correct_prediction = prediction.float() == y_train

accuracy = correct_prediction.sum().item() / correct_prediction.shape[0]

accuracy * 100
```

```

```