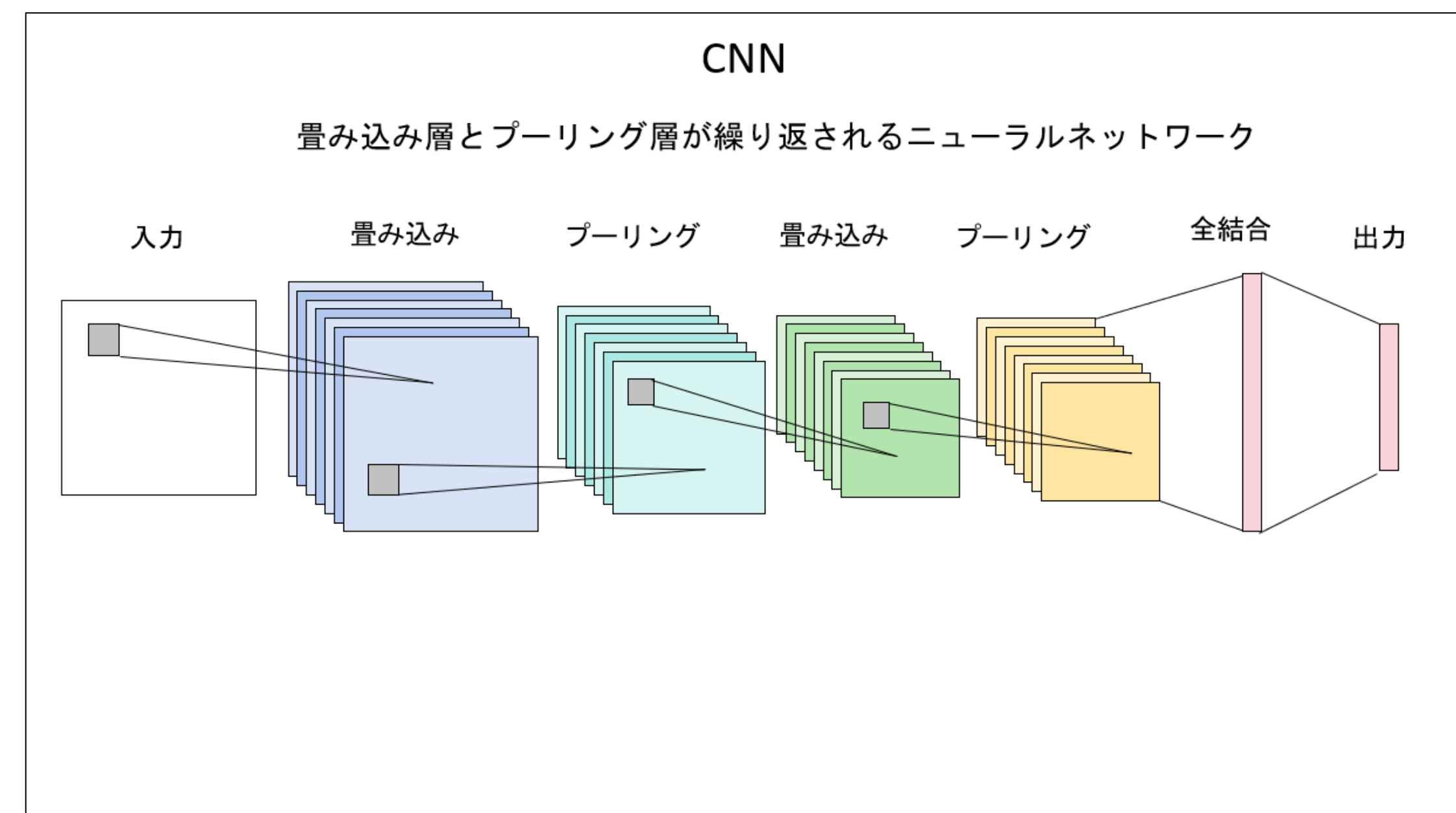
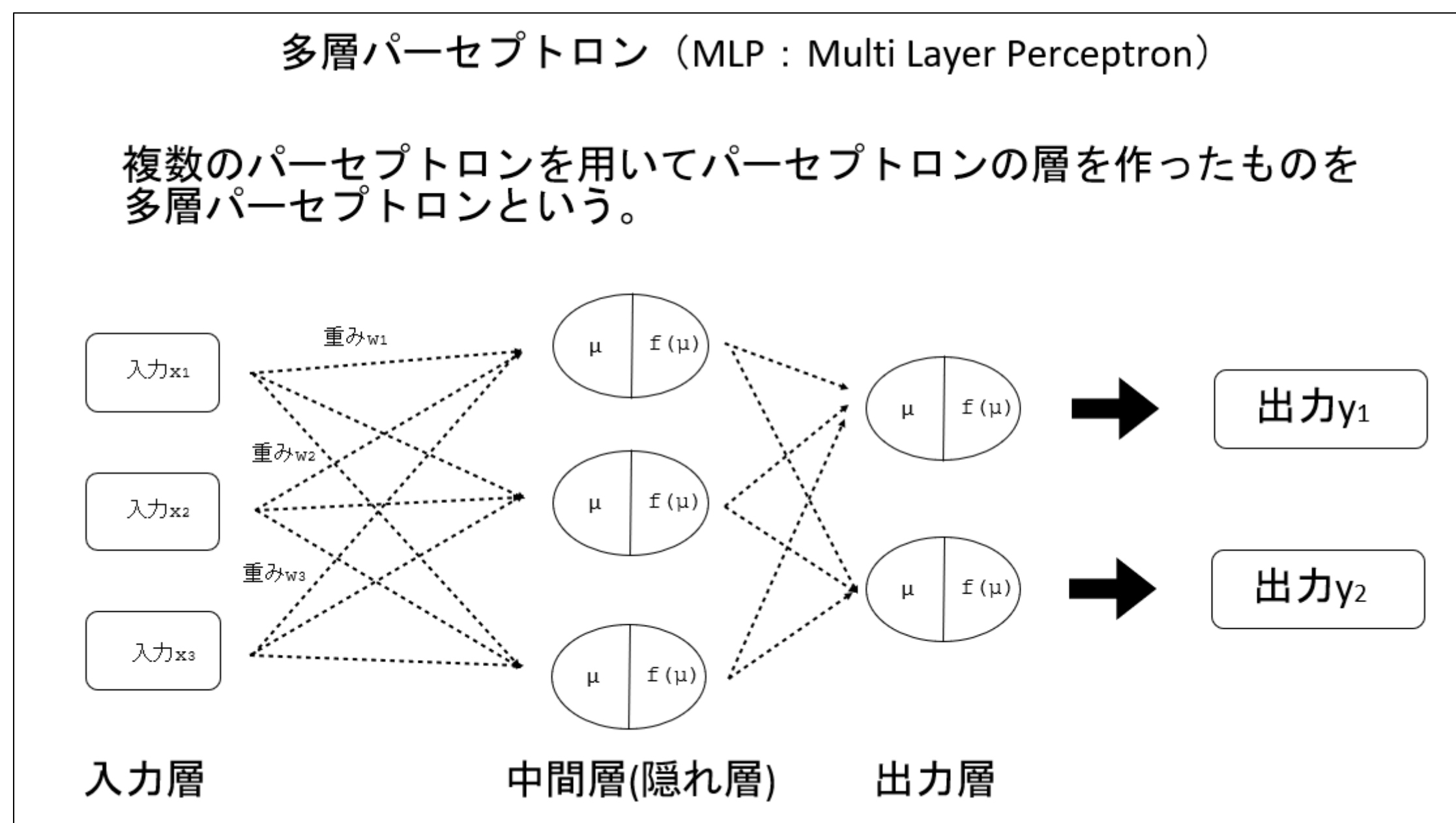


医療とAI・ビッグデータ応用

CIFAR10でMLP、CNNを実践

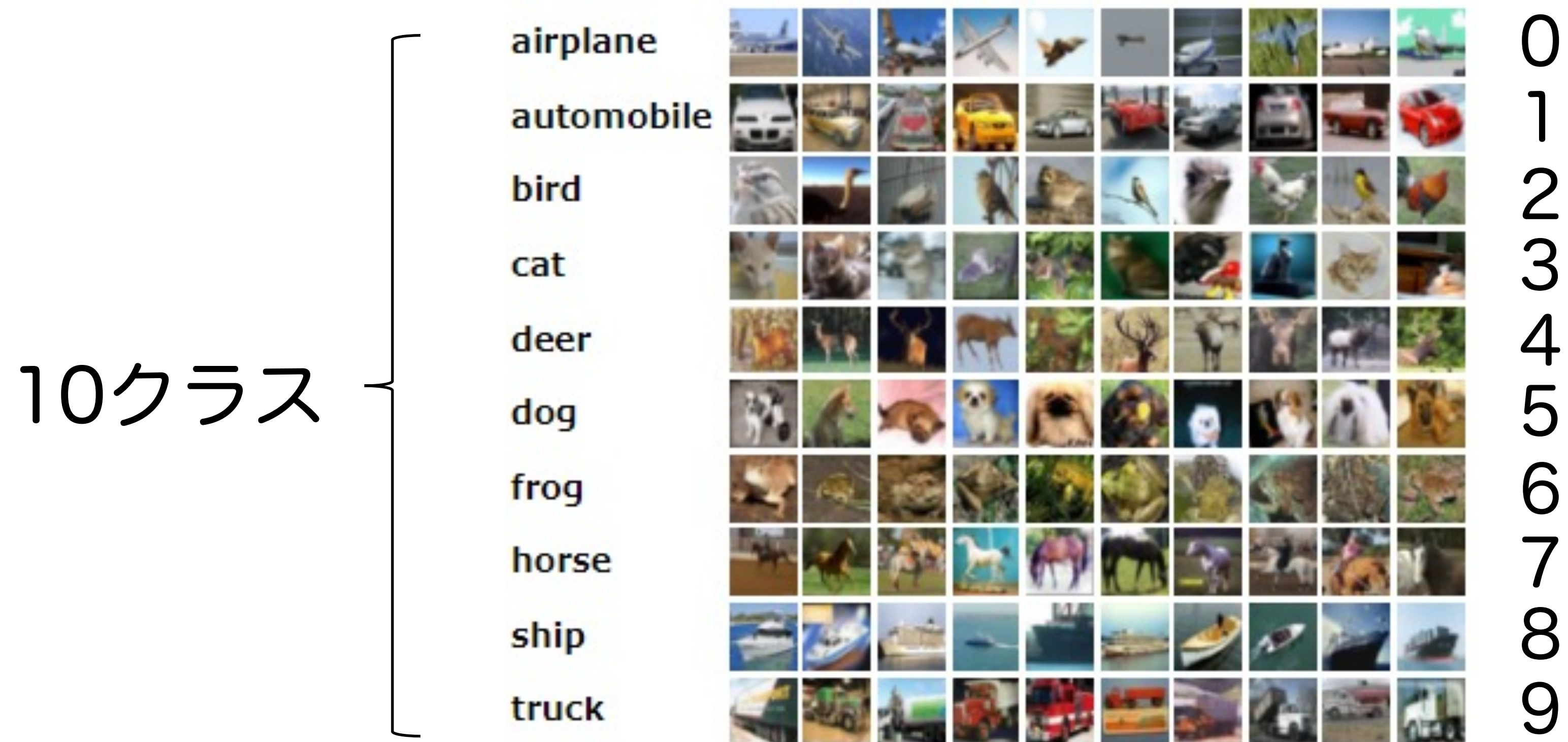
統合教育機構
須藤毅顕

これまでMNIST、FASHION-MNISTを使用して MLP、CNNを実践



グループに分かれて別のデータセットで試してみよう

CIFAR10



60000画像(50000画像：学習用、10000画像：テスト用)

MNIST、Fashion_MNISTと違う点はサイズとカラー

```
from keras.datasets import cifar10
```


グループ演習

ブレイクアウトルームでグループごとに分かれて演習を行ってもらいます。
ブレイクアウトルームの部屋番号をグループ番号とします。
原則グループで1つの答案を作ってそれぞれ全員に提出してもらいます。

問1~3: 画像の読み込みと加工

問1 testの画像の9991から10000番目の画像が、airplane、automobile、bird、cat、deer、dog、frog、horse、ship、truckのどれに相当するか調べなさい
(9991から10000番目まで順番に表示してください)

問2 trainの画像の最初の6枚を画像の上にplt.titleを使用して正解も表示してください

問3 trainの画像の最初の18枚をmatplotlibのsubplotを使って縦3、横6で表示する
その際に目盛りを消して表示してください。
(matplotlibで検索するとオプションの使い方が色々出てきます。) 公式URL
<https://matplotlib.org/>

問4 MLPを実行して一番testデータのaccuracyが高いモデルの学習結果を提出しなさい

問5 CNNを実行して一番testデータのaccuracyが高いものを提出しなさい

最後に全てコードを残して、**グループ番号(半角)_名前_学籍番号(半角).ipynb**で提出すること
(ex. 3_須藤毅顕_12345678.ipynb)

グループ演習

ブレイクアウトルームでグループごとに分かれて演習を行ってもらいます。
ブレイクアウトルームの部屋番号をグループ番号とします。
原則グループで1つの答案を作ってそれぞれ全員に提出してもらいます。

問1 testの画像の9991から10000番目の画像が、airplane、automobile、bird、cat、deer、dog、frog、horse、ship、truckのどれに相当するか調べたい
下のような出力結果が得られるように下記プログラムを完成させなさい

```
label = ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']  
for i in range( ① , ② ):  
    print( ③ ,label[ ④ ])
```

出力結果

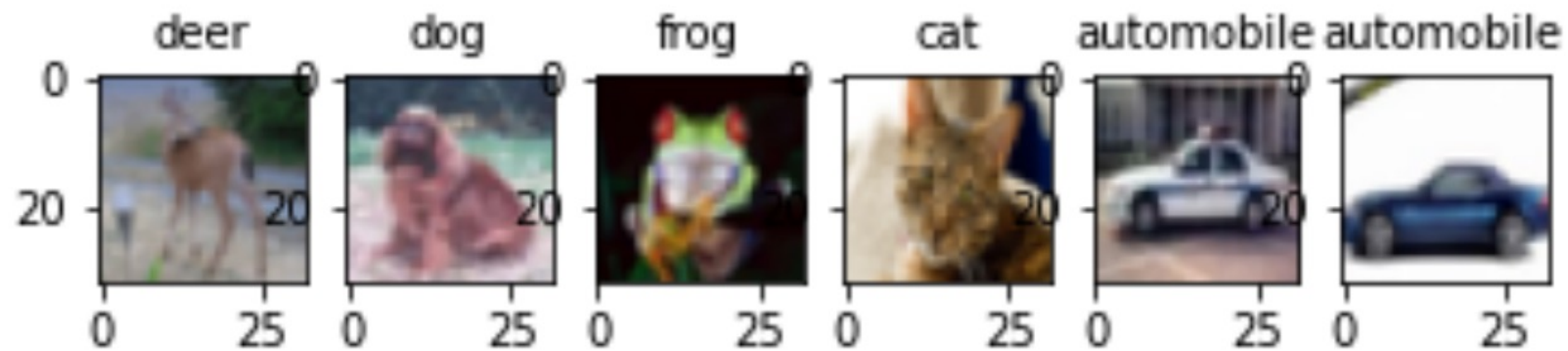
```
9991 horse  
9992 airplane  
9993 cat  
9994 dog  
9995 cat  
9996 ship  
9997 cat  
9998 dog  
9999 automobile  
10000 horse
```

グループ演習

ブレイクアウトルームでグループごとに分かれて演習を行ってもらいます。
ブレイクアウトルームの部屋番号をグループ番号とします。
原則グループで1つの答案を作ってそれぞれ全員に提出してもらいます。

問2

trainの画像の最初の6枚を画像の上にplt.titleを使用して正解も表示してください



例：正解の画像ではないです

plt.title(???,fontsize=??)で表示するサイズが変更されます

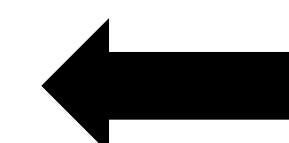
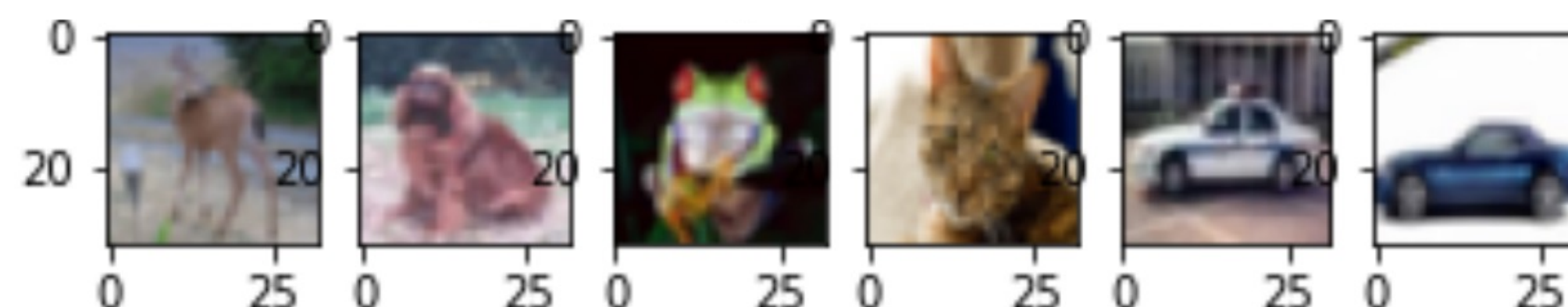
グループ演習

ブレイクアウトルームでグループごとに分かれて演習を行ってもらいます。
ブレイクアウトルームの部屋番号をグループ番号とします。
原則グループで1つの答案を作ってそれぞれ全員に提出してもらいます。

問3

trainの画像の最初の18枚をmatplotlibのsubplotを使って縦3、横6で表示する
その際に目盛りを消して表示してください。
(matplotlibで検索するとオプションの使い方が色々出てきます。)

公式URL
<https://matplotlib.org/>



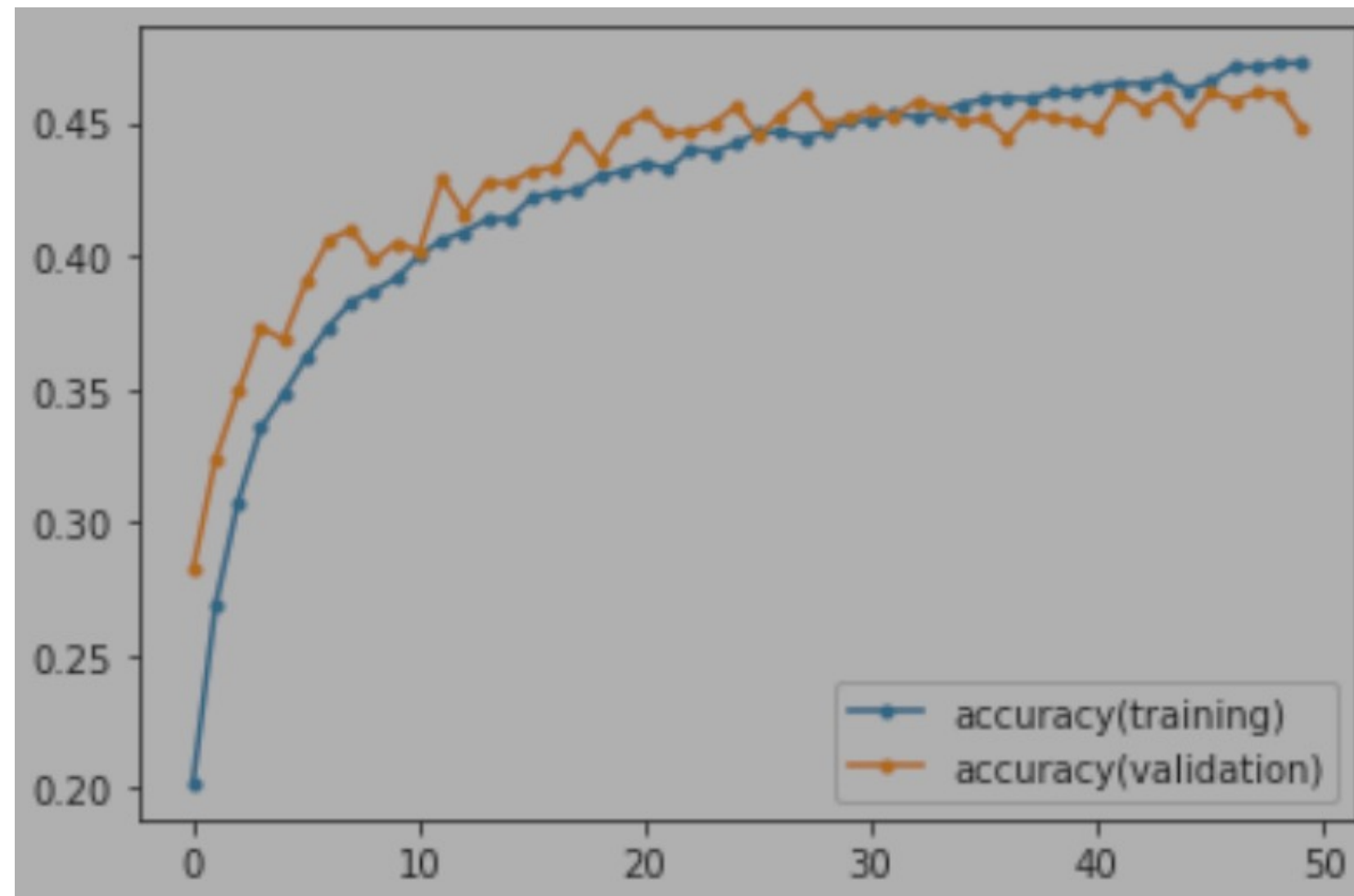
この目盛りを消してください

問4 MLPを実行して一番testデータのaccuracyがグループ内で高いモデルの学習結果を提出しなさい

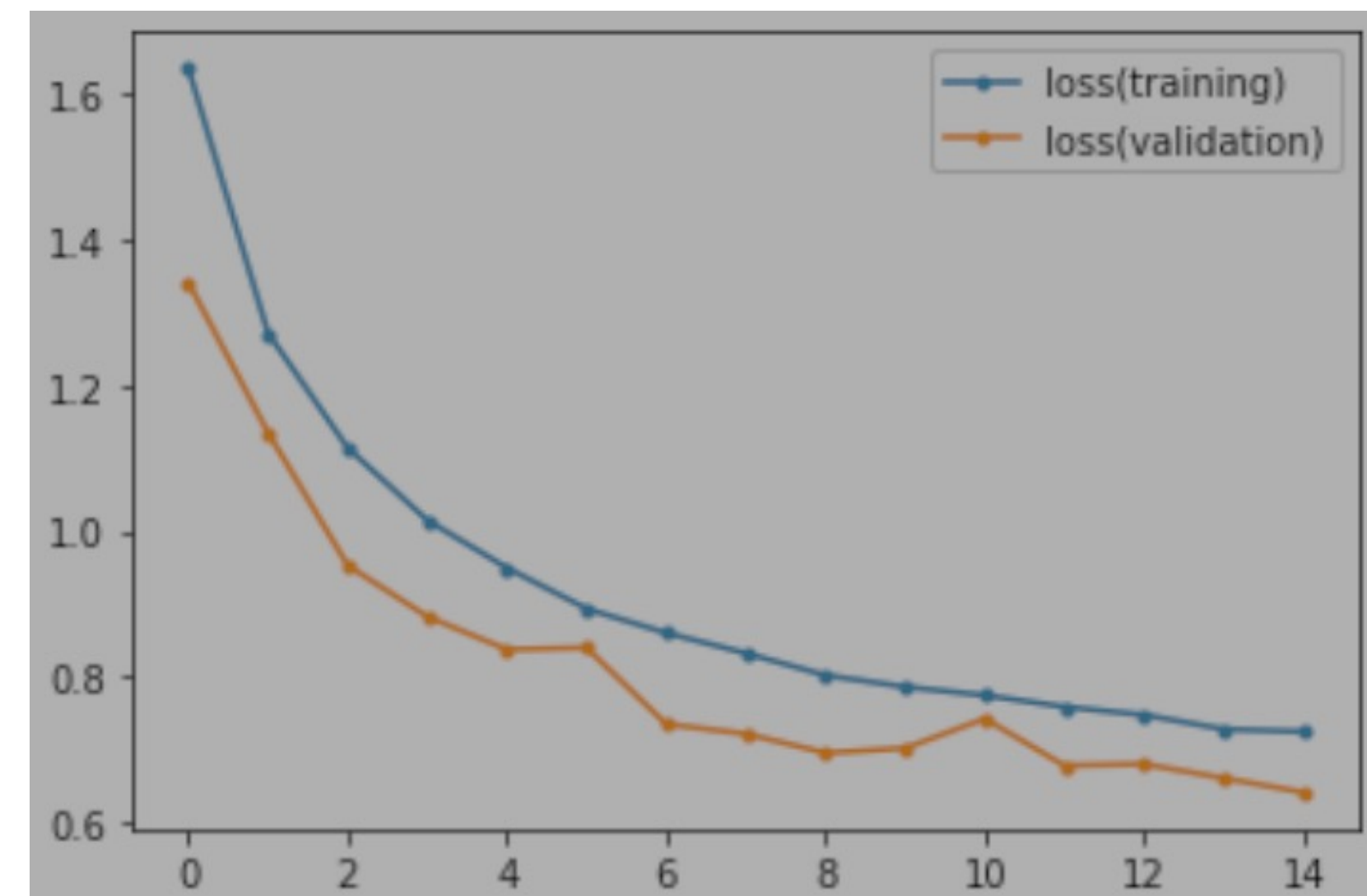
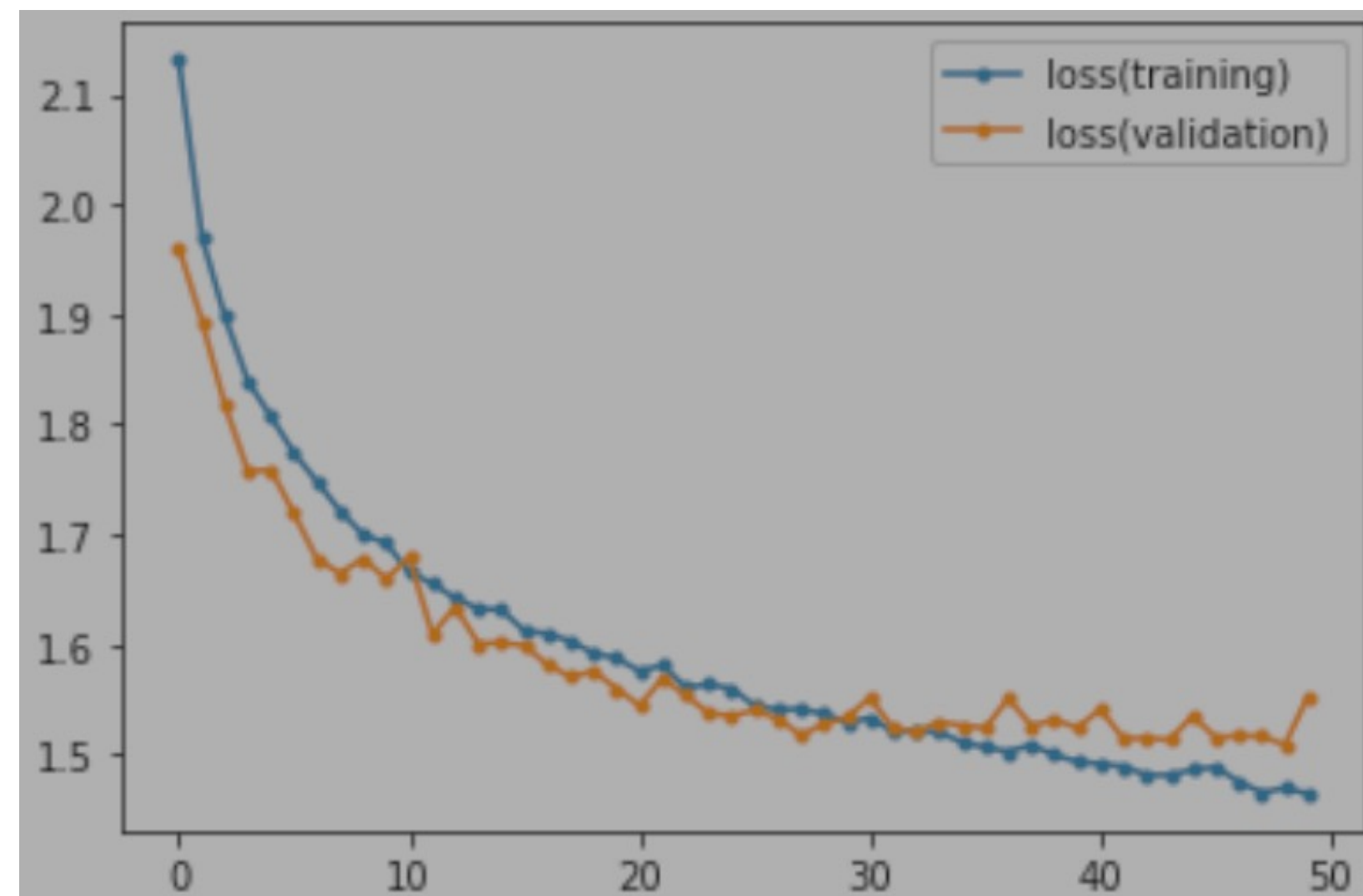
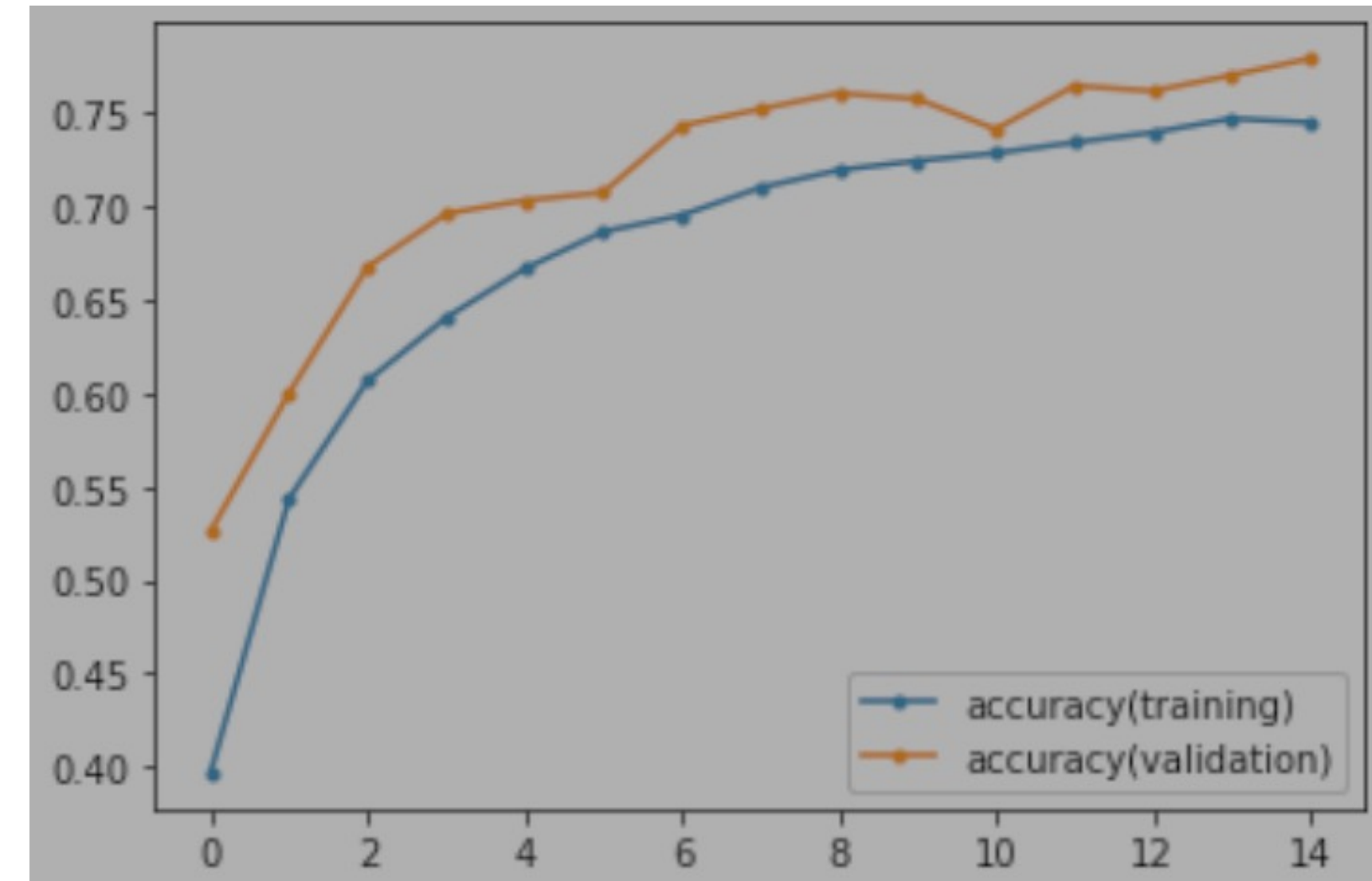
問5 CNNを実行して一番testデータのaccuracyがグループ内で高いモデルの学習結果を提出しなさい

最後に全てコードを残して、**グループ番号(半角)_名前_学籍番号(半角).ipynb**で提出すること
(ex. 3_須藤毅顕_12345678.ipynb)

試しにやったMLPでは精度45%でした



試しにやったCNNではエポック15
で精度75%ぐらいでした



- ・ 1グループ6～7人（計29～25グループ）
- ・ 取り組み方は自由
 - ・ グループとして1つの答えにしていもらい、各自が提出
 - ・ MLPとCNNの学習時(特にCNN)が時間がかかるので、手分けすることを推奨
 - 案1) みんなでモデルを作る、1～2人のPCで実行(一番サクサク動くPCを探す?)
待つ間に画像の表示方法を考える
学習中の学生は調べものをする、提案するなど
 - 案2) みんなで一斉に取り組む、実行する際にグループ内でモデルを調整して比較しあう

問4, 5は時間内で一番良い結果のものを提出してもらっても勿論大丈夫ですし、
時間外にさらに取り組んでももらっても大丈夫です。
期限はいつもの課題同様、翌週の日曜日までとします。
精度が高いグループには加点されます。
※全員が内容を理解するように努めてください

時間外になる可能性もあるのでグループ内の連絡先は交換しておきましょう

次回の応用9は来週月曜日にアップロードします
(課題期限は1/15までです)

補足説明

画像の表示

```
plt.imshow(画像データ,'gray')
plt.imshow(画像データ)
```

白黒
カラー

```
import matplotlib.pyplot as plt
plt.imshow(x_train[0], 'gray')
plt.show()
```

画像を描画する

matplotlib(描画ライブラリ)

'gray'で白黒を指定

数字の5らしい

plt.subplot(2,5,i+1)にすると縦2、横5の図になる

```
for i in range(10):
    plt.subplot(2,5,i+1)
    plt.imshow(x_train[i], 'gray')
plt.show()
```

補足説明

```
test = [1,2,3,4,5]
print(test)
print(test[0]) # 1つ目の1が出力されます
test2 = [[1],[2],[3],[4],[5]]
print(test2)
print(test2[0]) # 1つ目の[1]が出力されます
print(test2[0][0]) # [1](test2[0])の1つ目である1が出力されます
test3 = [[1,2,3],[4,5,6],[7,8,9],[10,11,12],[12,13,14]]
print(test3)
print(test3[0]) # 1つ目の[1,2,3]が出力されます
print(test3[0][0]) #[1,2,3](test3[0])の1つ目である[1]
```

配列の要素の取り出し方

```
import numpy as np
test = np.array([1,2,3,4,5])
print(test)
print(test[0])
test2 = np.array([[1],[2],[3],[4],[5]])
print(test2)
print(test2[0])
print(test2[0][0])
test3 = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12],[12,13,14]])
print(test3)
print(test3[0])
print(test3[0][0])
```

リストも**numpy**配列も
同じように取り出すこと
が出来ます

1次元の変換と正規化

`x_train = x_train.reshape(x_train.shape[0], 縦のサイズ×横のサイズ×チャンネル数)/255`

白黒 `x_train = x_train.reshape(x_train.shape[0], 縦のサイズ×横のサイズ×1)/255`

MLPの授業では白黒だったので×1を省略しました

カラー `x_train = x_train.reshape(x_train.shape[0], 縦のサイズ×横のサイズ×3)/255`

深層学習前のデータの整理

x_train (特徴量)

- ・ 画像の2次元の配列を1次元にする

まだ入力なくていいです

```
x_train = x_train.reshape((x_train.shape[0], 784))
x_test = x_test.reshape((x_test.shape[0], 784))
print(x_train.shape)
print(x_test.shape)
```

```
(60000, 784)
(10000, 784)
```

← この784は $28 \times 28 \times \mathbf{1} = 784$

質問などは随時受け付けます。

時間内にグループで回答が終わり、他のグループのサポートが出来る学生がいればご連絡下さい。
(成績にも加味致します。)

~~この授業は欠席した学生は課題の提出も無しとなります。~~

この授業コマのグループワークに出席できなかった学生は、
個人で演習を実施して課題を提出してください。
グループ番号は半角のゼロ「0」としてください。