1. 请问这里的 X 为什么需要除以 255，是用于 normalize 么

```
# Wash the raw data to make it the same as assignment-
# and normalize the all the image vectos to 0 and 1

X = x_train.reshape(60000, -1)
X = X/255
y = y_train
print(X.shape)
print(y_train.shape)
```

写第一次作业的时候我发过一个邮件问老师
要不要 renormalize data set 到 [0,1], 老师说自己
判断。。。。。。

然后这次作业第一题 perceptron 要求 renormalize,
第三题KMEAN 我就也 renormalize 了,
理论上这题 做不做 renormalzie 没影响,
但是 第一次作业鸢尾花那题，老师应该给一个说明的
花蕊的 length，width 怎么 renormalize 影响还挺大的

I. Perceptron (30 Marks)

For this part you will write code to implement perceptrons and the perceptron learning algorithm.

Perceptrons: You will train 10 perceptrons that will, as a group, learn to classify the handwritten digits in the MNIST dataset. See the class slides for details of the perceptron architecture and perceptron learning algorithm. Each perceptron will have 785 inputs and one output. Each perceptron target is one of the 10 digits.

Preprocessing: Scale each data value to be between 0 and 1 (i.e., divide each value by 255, which is the maximum value in the original data). This will help keep the weights from getting too large. Randomly shuffle the order of the training data.

2. 这里为什么需要 subpress warning

```
# this is the funciton requried by the assignemnt,
# it takes two requried input, X, a nd-array of size N
# it also takes a optional argument input, P, the max
# it has 3 returns in a list
# the 1st return is a list of N labels represent each
# the 2nd return is a dictionary, key is the label of
# the 3rd return is a series of SSE (J^Cluster) for ea
def k_mean_minst_clustering(X, K, P=30):
  real_ep_max = P

  # sub_press any warning message
  import warnings
  warnings.filterwarnings("ignore")

  label_list_fro_each_images = []
  list_of_list_of_clusters = [[] for _ in range(K)]
  Jcluster = []
```

好问题，当时我有一个 runtime warning，
赶着交作业我就糊弄过去了

/usr/local/lib/python3.7/dist-packages/sklearn/cluster/_kmeans.py:1146: RuntimeWarning: Explicit initial center position passed: performing only one init in KMeans instead of n_init=10. self._check_params(X)

```
# train it over P times:
for _ in range(1, real_ep_max+1, 1):
  # set the center position and train the model
  kmeans = KMeans(n_clusters=K, init = initial_centers, random_state  = 500, tol = 0.001, max_iter=1)  #
  mykmean = kmeans.fit(X)
  initial_centers = copy.deepcopy(kmeans.cluster_centers_)
```

我们可以讨论一下，这个地方其实我也不太懂 到底应该怎么写，我就用 一个 trick 糊弄过去了
我们的目标是在 O（N） 的时间复杂度下 plot error vs epoch_num 哈

我 最开始的思路 是 new 一个 kmean instance，然后限定步长是 1，
然后 train 30 次，每次记录 error

```
kmeans = KMeans(n_clusters=K,  random_state=0,tol = 0.001, max_iter=1)

# train it over P times:
for _ in range(1, real_ep_max+1, 1):
  # set the center position and train the model
  kmeans = kmeans.fit(X)
  # initial_centers = copy.deepcopy(kmeans.cluster_centers_)

  # print(mykmean.inertia_)
  # centers = mykmean.cluster_centers_
  # upodate the J_cluseter
  SSE = kmeans.inertia_
  Jcluster.append(SSE)
```

但是这样画出来的图，error 是 不随着 epoch_num 减小的，我当时就匆匆读了一下 API，说 fit() 返回一个 fitted estimator
但是这个 fitted estimator 的 initial value 是没有update 过的，所以我就只好在 loop 里面 每次 new 一个 instance， 额外给一个 init value，但是这样做的弊端是 会有 warning，我不确定是我没理解 skrlearn Kmean 这个 class 的 design pattern 还是

它们这个 class 设计的有问题。。。
你要是有更好的办法请告诉我

fit(*X*, *y=None*, *sample_weight=None*) [source]

Compute k-means clustering.

Parameters:

X : *{array-like, sparse matrix} of shape (n_samples, n_features)*
Training instances to cluster. It must be noted that the data will be converted to C ordering, which will cause a memory copy if the given data is not C-contiguous. If a sparse matrix is passed, a copy will be made if it's not in CSR format.

y : *Ignored*
Not used, present here for API consistency by convention.

sample_weight : *array-like of shape (n_samples,), default=None*
The weights for each observation in X. If None, all observations are assigned equal weight.

New in version 0.20.

Returns:

self : *object*
Fitted estimator.

3. 这里计算 SSE 不应该是 mykmean.inertia_ / N 么

```
# train it over P times:
for _ in range(1, real_ep_max+1, 1):
  # set the center position and train the model
  kmeans = KMeans(n_clusters=K, init = initial_centers, random_state  = 5
  mykmean = kmeans.fit(X)
  initial_centers = copy.deepcopy(kmeans.cluster_centers_)

  # print(mykmean.inertia_)
  centers = mykmean.cluster_centers_
  # upodate the J_cluseter
  SSE = mykmean.inertia_
  Jcluster.append(SSE)
```

Cost Function (Error Function)

- The goal is to minimize cost function J w. r. t

$$J(b,m) = \sum_{i=1}^{n} (y_i - b - mx_i)^2$$

Linear regression goal → $min_{b,m} J(b,m)$

J is a sum of squares, a second order polynomial equation

我们第六次课 p48 页定义的 J 没有 除以N呢，这东西到底跟 inertia 有啥关系我也不知道，我感觉 J 就是 intertie 【作为学物理的，除不除常数不影响性质，哈哈哈哈哈】

4. 代码中用 lbs, dict_of_clusters, JvE, myKmeanModel 来保存 k_mean_minst_clustering(X,K=10,P=20) 返回的四个参数（下图一），这样的话 dict_of_clusters 和 myKmeanModel.cluster_centers_会有什么区别么？(下图二)

没有区别哈哈哈哈，当时着急赶due，就瞎写的，怎么方便就怎么来的。。。哈哈哈，见笑了。

但是你需要注意一点，重要的 数据，尤其是 class 你不熟悉的情况下，存取的时候都想着 make a defensive deepcopy

要不然，假如 Kmean 这个 class，它设计的时候用了 单例模式 （https://sourcemaking.com/design_patterns/singleton）

之后别人再用一下这个 class，你的数据就没了

这都是我自己的感觉哈，我也不是科班出身的，都是自己猜的，哈哈哈哈

我参考了下 https://stackoverflow.com/questions/61192374/interpreting-k-means-cluster-centers-output，xxxx.cluster_centers 应该是 sklearn.kmeans 内置的最后一次迭代后，输出所有 centroids 的功能。但我尝试运行你写的脚本，发现我这边一直卡在执行 k_mean_minst_clustering(X,K=10,P=20)方程这里，就只好问你啦

我刚才跑了一下，我的代码在 google colab 里还是可以跑的，我上学期写的时候，colab 还没有 SKLearn 1.x 的 lib，我需要自己从 github 里下载到colab虚拟机上，自己安装， 现在已经有了1.0.2 了

这个就是比较慢吧，我 p=30 大概要跑 30分钟我感觉，

你要是一直卡，可以看一下你的运行环境，你要是 local 跑的话，
pycharm 好像默认给 python 1 G 的内存，估计不够，你需要手动调高一点
你要是 debug 的话，可以降低 P，从5 开始，然后也不要train 所有的数据，比如 先 train 1/10，确保代码能跑先，然后再 train 所有的数据

5. 这里用 dist += (a[i]-b[i])*(a[i]-b[i]) 而非 dist += (a[i]-b[i])**2；是有什么特殊意义么？

Now, need to calculate the top 10 nearest images to each center

```
]:
# Now, need to calculate the top 10 nearest images to each center
near_ten_elements = []
mycenterpoints = myKmeanModel.cluster_centers_

def cal_dist(a,b):
  dist = 0
  for i in range(len(a)):
    dist += (a[i]-b[i])*(a[i]-b[i])
  return dist
```

就是因为我单纯的记不住 Java， python， numpy，matlab 各自的乘方公式。。。。
这个又肯定是对的。。。。。哈哈哈哈哈，我猜测跟**2 比不会有效率上的损失。

刚才测了一下，好像 明着写会快好多。。。。我也不知道为啥，我猜测明着写method 不用进 stack

怎么乘方不重要，但是装饰器你要是不熟悉，建议你学一下

https://refactoring.guru/design-patterns/decorator

```
@GetRunTime
def longPower(n):
  for i in range(n):
    result = i*i
  return result

@GetRunTime
def shortPower(n):
  for i in range(n):
    result = i**2
  return result

@GetRunTime
def powerPower(n):
  for i in range(n):
    result = math.pow(i,2)
  return result


TEST_EPOCH = 10000000
print(longPower(TEST_EPOCH))
print(shortPower(TEST_EPOCH))
print(powerPower(TEST_EPOCH))
```

```
longPower函数运行时间为1.3384368419647217
99999980000001
shortPower函数运行时间为3.342186689376831
99999980000001
powerPower函数运行时间为2.055995464324951
99999980000001.0
```