# NCTU 2019 Computer Vision

# HW5 Image Classifier

Hsin-Yu Chen, Yuan-Syun Ye

May 2019

## 1 Introduction

In this homework, we need to builds a classifier to categorize images into one of 15 scene types. We must complete the task by three ways: Tiny images representation + nearest neighbor classiﬁer; Bag of SIFT representation + nearest neighbor classiﬁer; Bag of SIFT representation + linear SVM classiﬁer. That is, we need to Implementation two kinds of images representation with KNN classiﬁer and linear SVM classiﬁer. Also, we tried standard CNN model and ResNet with pre-train model to classify the images. We Implement task1 and task2 in both MATLAB and python, and task3 in MATLAB.

## 2 Implementation

#### 2.1 images representation

#### 2.1.1 Tiny images representation

In Tiny images representation, all we need to do is resize the image to 16\*16. Then because we need to compute the distance for nearest neighbor classiﬁer, we reshape the image into shape 1\*256.

|  |  |
| --- | --- |
| Figure 1 Tiny image (MATLAB) | Figure 2 Tiny image (Python) |

#### 2.1.2 Bag of SIFT representation

We use SIFT to find the descriptors for each image, then stack all the descriptors in a numpy array. Then we do k-means clustering to find out k clustering center from all the descriptors, where we set k = 300. Next, we classify features of each image and calculate the histogram of them. Then the histogram is the Bag of SIFT representation of the image.

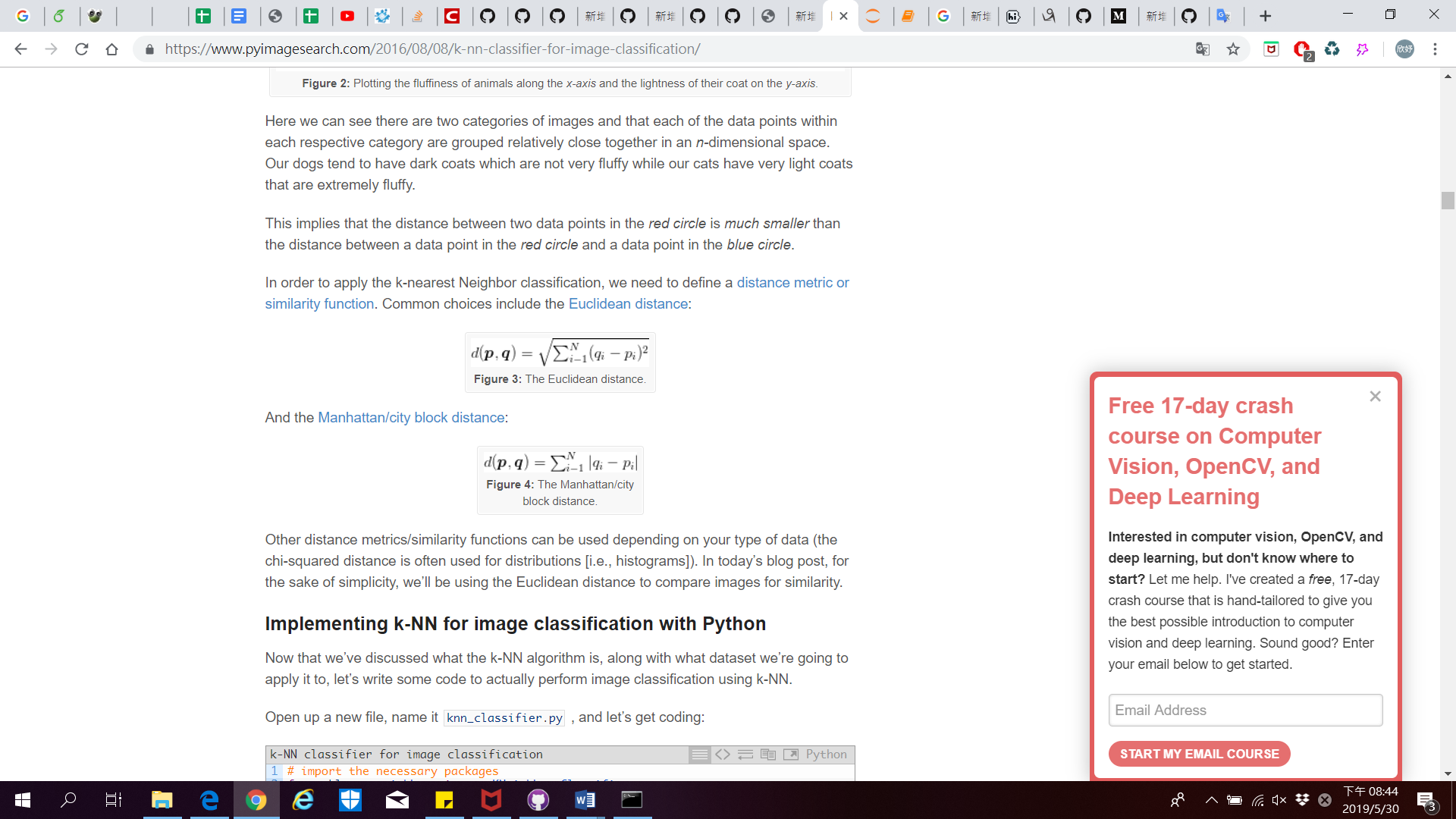
|  |  |
| --- | --- |
| Figure 3 Find K clustering center | Figure 4 Calculate the histogram of image features |

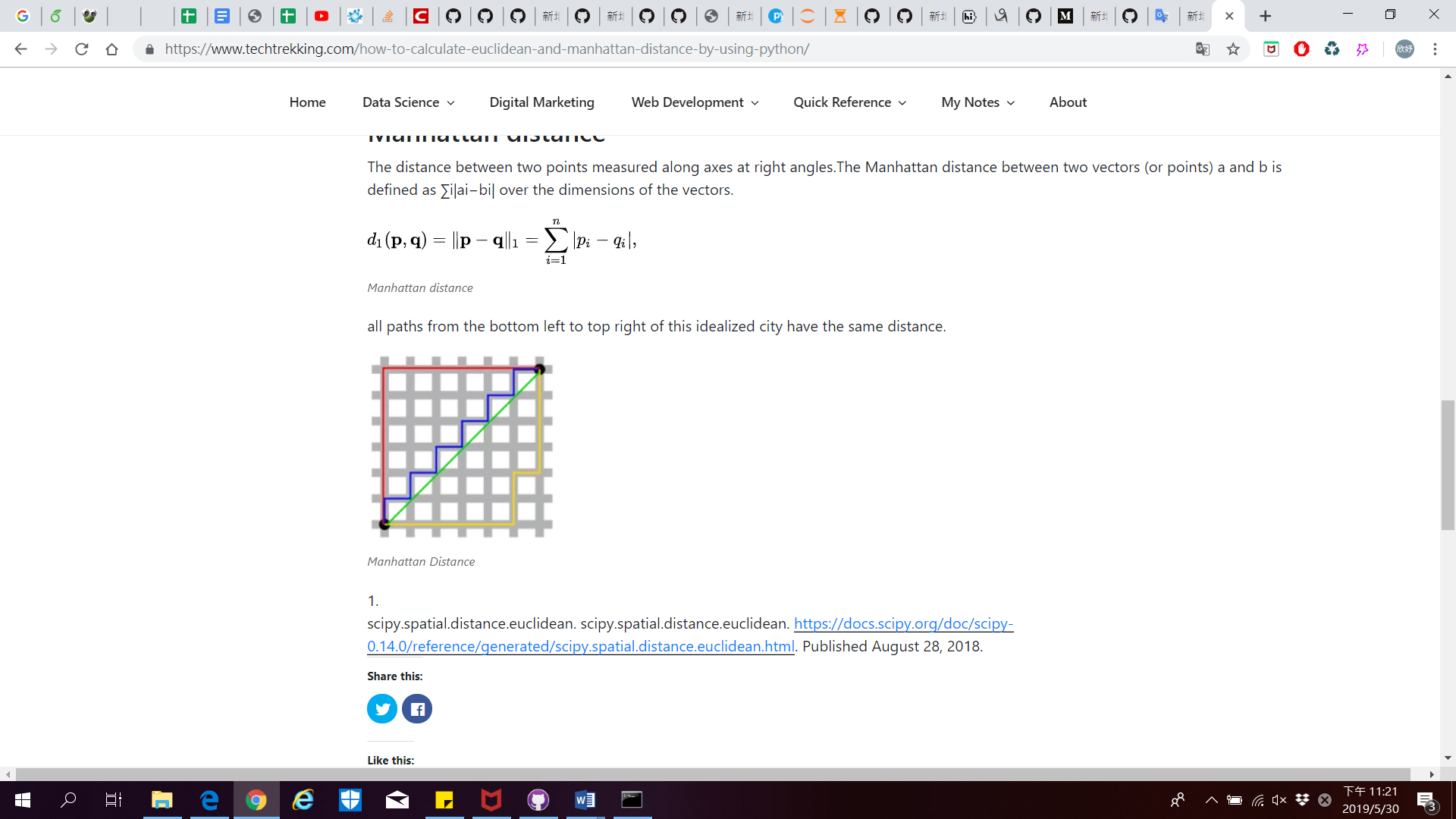
|  |  |
| --- | --- |
| Figure 5 The implementation of Bag of SIFT representation (Python) |  |

### 2.2 Classiﬁer

#### 2.2.1 Nearest neighbor classiﬁer

We Implementation both Euclidean distance and Manhattan distance to compute distance between the test image and each training data.

Euclidean distance 

Manhattan distance 

Then setting the label of test image as same as the training data which has shortest distance with the test image.

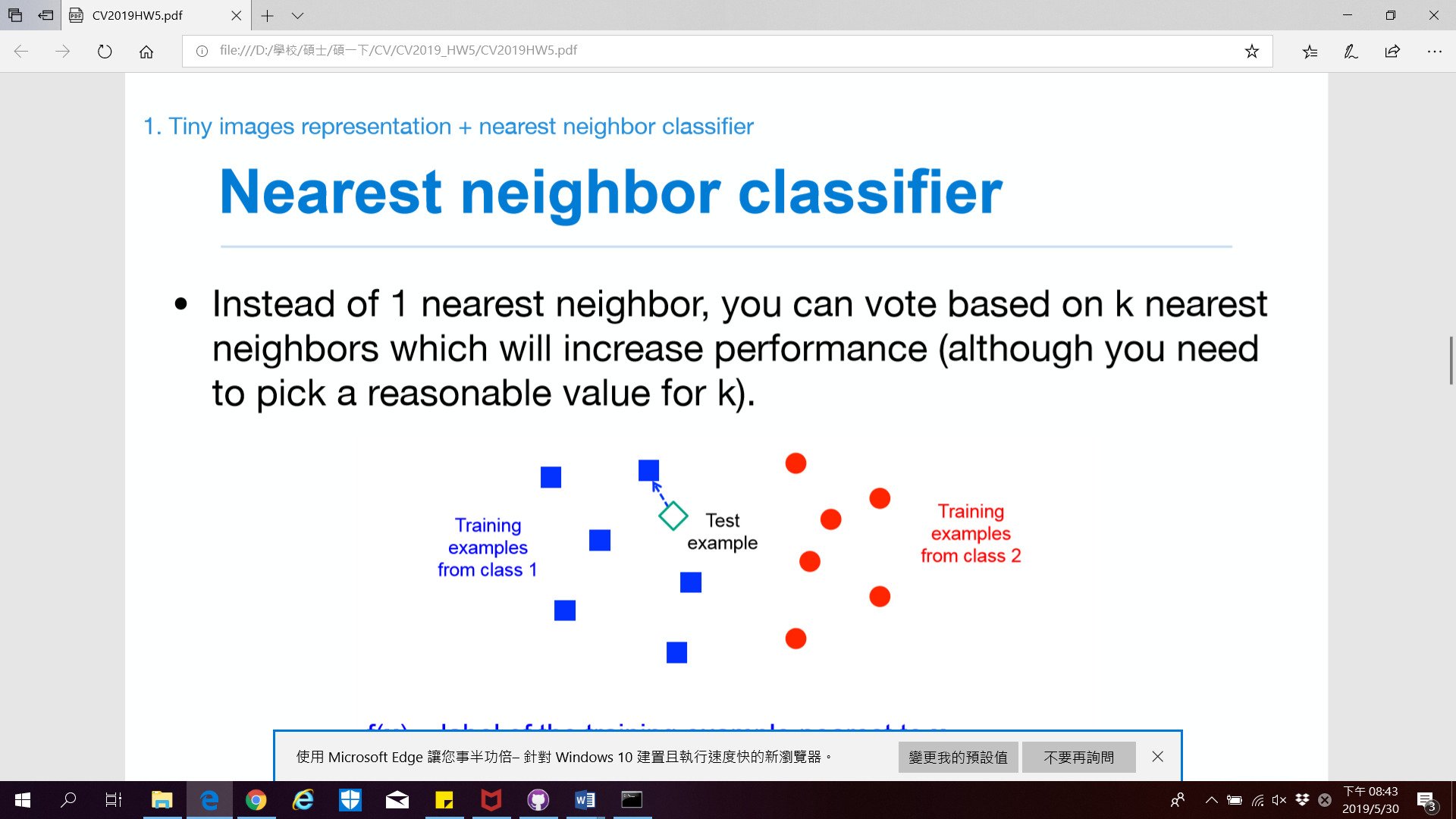
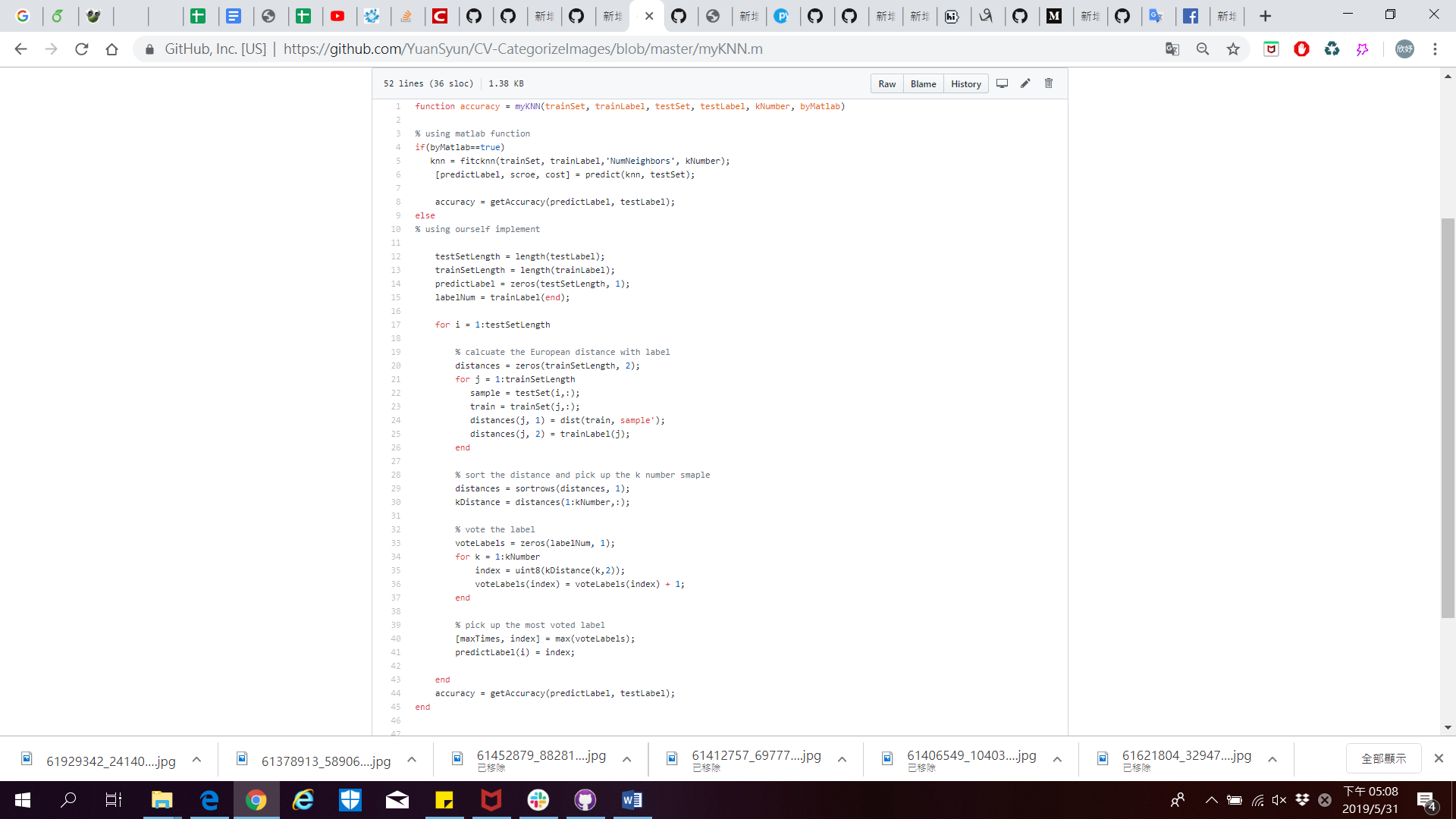
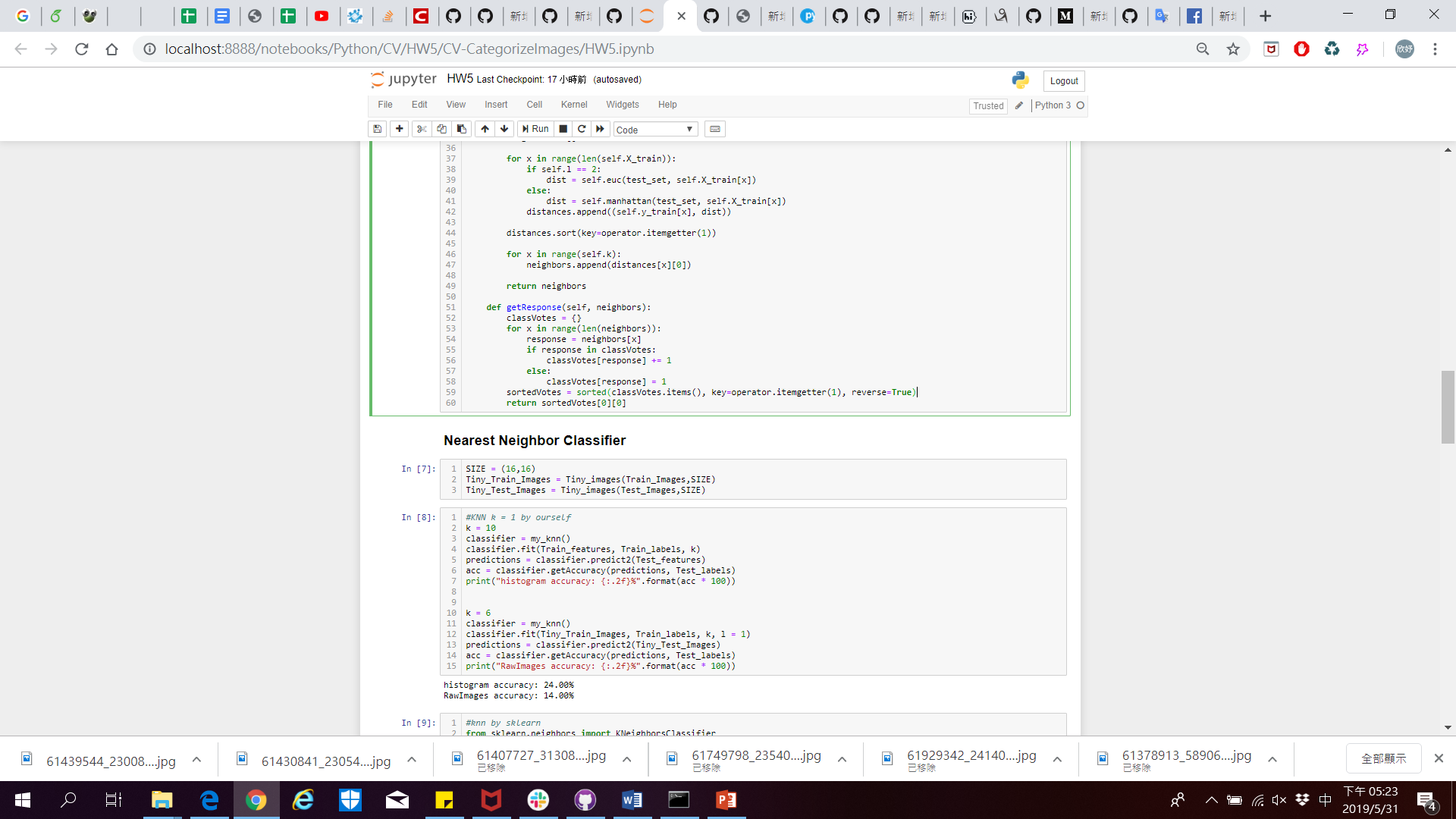
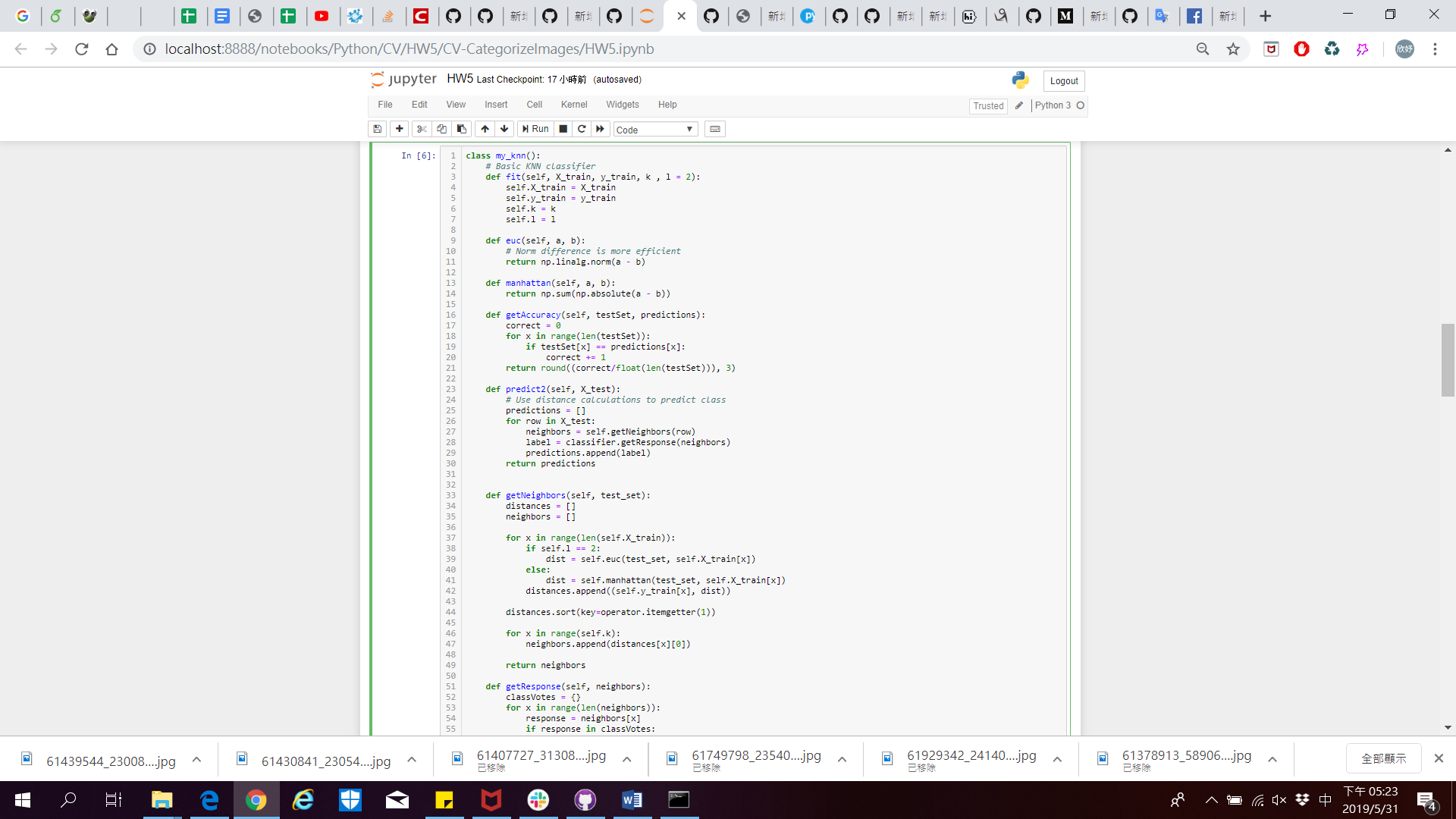


Figure 6 Set the test sample label to class one





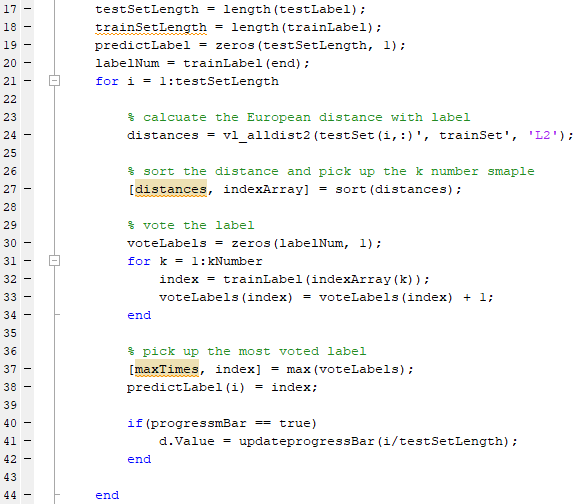
****

Figure 7 Nearest Neighbor Classifier (Matlab Code)

#### 2.2.2 Linear SVM classiﬁer

### 2.3 Deep Learning

#### 2.3.1 Standard CNN model

We use 2 layer CNN model with Adam as optimizer, and Cross Entropy as loss function to train the model.

#### 2.3.2 Pre-train ResNet model

Because the training dataset are too small and the accuracy doesn’t seem good, we constructs a pre-train ResNet-34 model. The model has four layers with Adam as optimizer, and Cross Entropy as loss function.

## 3 Experimental

### 3.1 Tiny images representation + nearest neighbor classiﬁer

Accuracy: 21.5%(k=5, L1 Distance), 18.5%(k=5, L2 Distance), 18.5%(Matlab function)

|  |  |  |
| --- | --- | --- |
| Figure 8 L1 distance | Figure 9 L2 distance | Figure 10 MATLAB function |

### 3.2 Bag of SIFT representation + nearest neighbor classiﬁer

Accuracy: 35%

|  |  |  |
| --- | --- | --- |
| Figure 11 SIFT Features |  | Figure 12 using Matlab KNN |

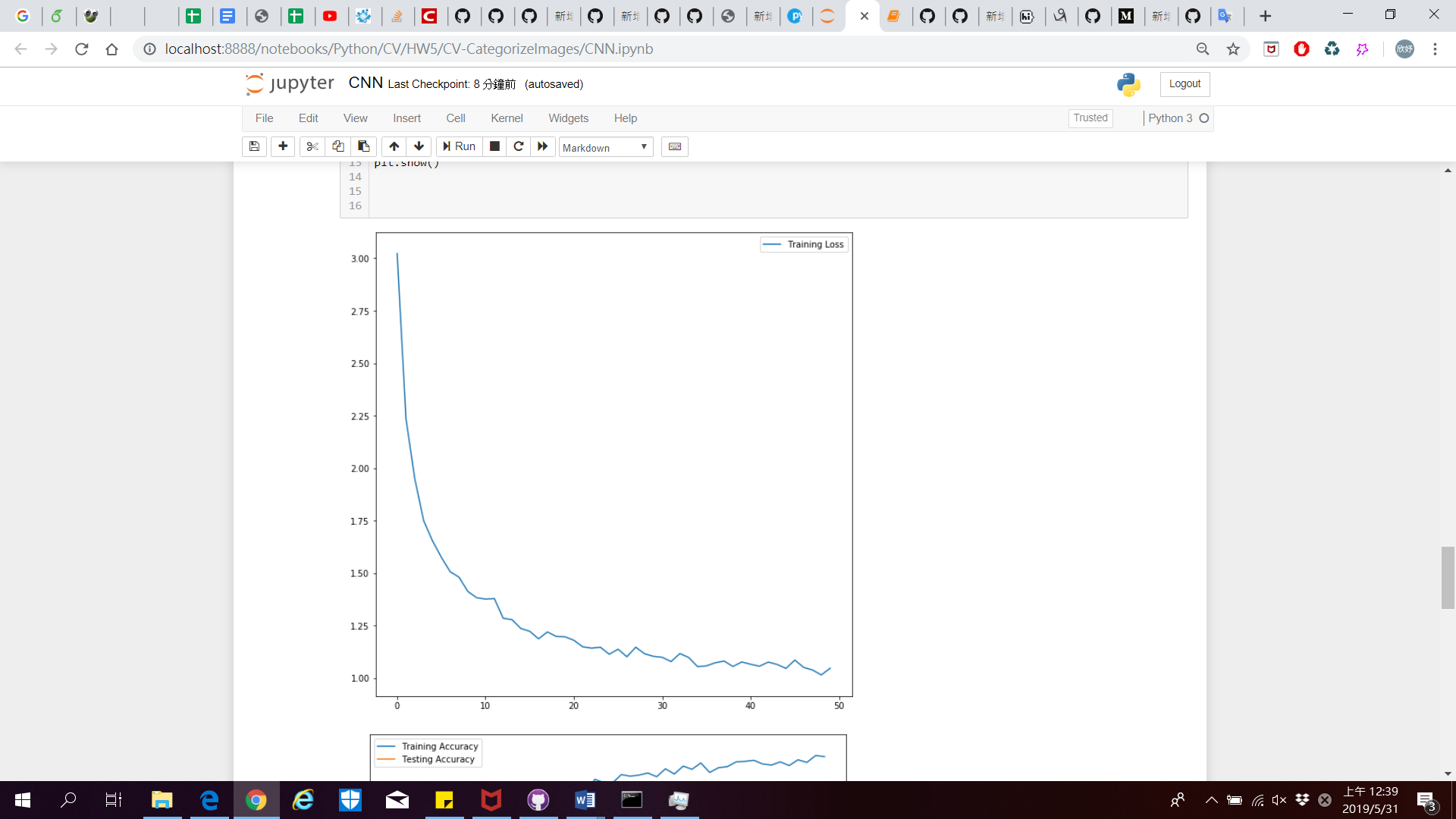
### 3.3 Bag of SIFT representation + linear SVM classiﬁer

Accuracy: %

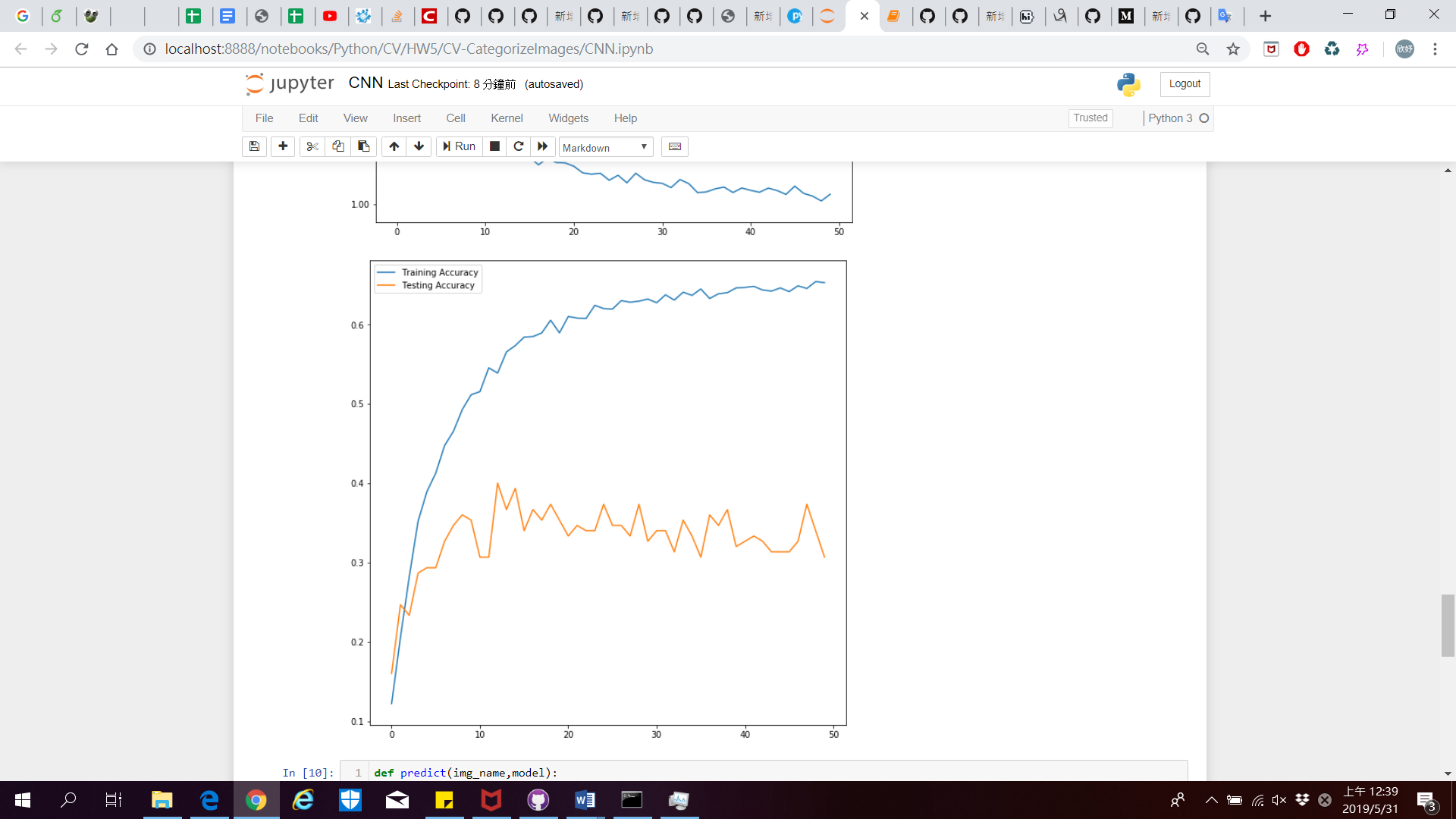
### 3.4 Deep Learning by standard CNN model

Accuracy: 30-35%

Because the training dataset are too small and the accuracy doesn’t seem good



🡺 Training Loss



🡺 Training Accuracy & Test Accuracy

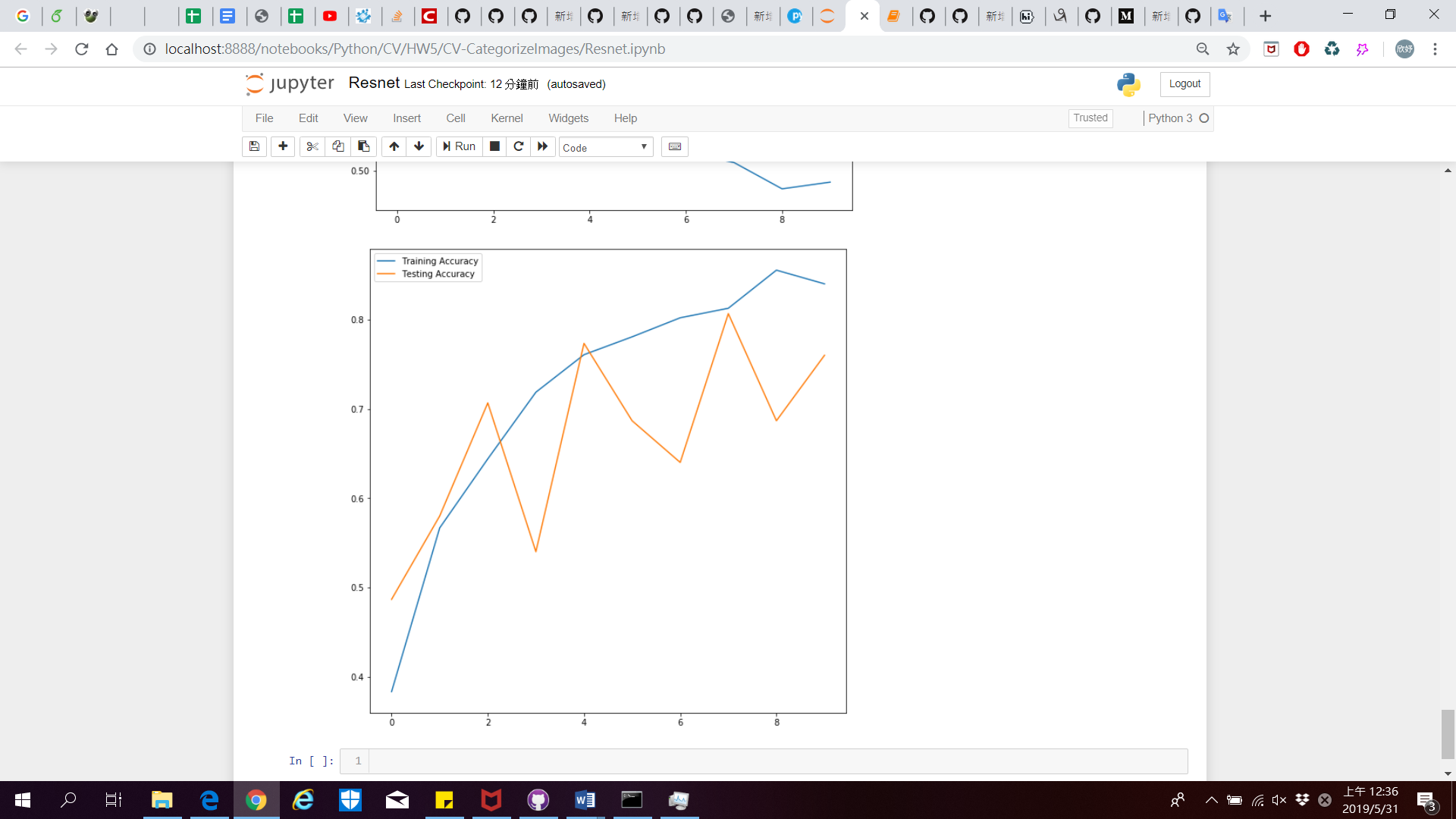
### 3.5 Deep Learning by pre-train ResNet model

Accuracy: 75-80%

Due to the limitation of GPU memory size, we only run 10 epoch. But because we use the pre-train model, that has a good test accuracy.



🡺 Training Loss



🡺 Training Accuracy & Test Accuracy

## 4 Discussion

In task1, Tiny images representation + nearest neighbor classiﬁer, we found out that when compute distance between the test image and each training data, accuracy of Manhattan distance is better than Euclidean distance. That may because gray value of Tiny image is between 0 to 255, and if we used Euclidean distance, the distance would become large for some extreme case which may Influence the result.

## 5 Conclusion

We Implement image classifier in two different ways with two kind of image representation. And we also Implement two deep learning model, CNN and ResNet.

Then we show the accuracy in each task and the value of k for knn classiﬁer.

## 6 Work Assignment Plan

This homework divided into two parts. Yuan-Syun Ye is responsible for the part of cording by MATLAB and checks this report. Hsin-Yu Chen is responsible for the part of cording by python and the writing of the report.

## References

https://www.pyimagesearch.com/2016/08/08/k-nn-classifier-for-image-classification/