陽明交大
NYCU

物聯網裝置與平台
IoT Devices and Platforms

曾煜棋、吳昆儒

National Yang Ming Chiao Tung University

	日期	主題
1	9/17	(加退選9/13-27) 課程介紹, arduino簡介
2	9/24	物聯網裝置: Arduino basic introduction
3	10/1	物聯網裝置: Arduino Digital Interface
4	10/8	物聯網裝置: Arduino Analog Interface
5	10/15	sensor介紹 part 1
6	10/22	sensor介紹 part 2
7	10/29	sensor介紹 part 3
8	11/5	(期中考周11/1-5) sensor介紹 part 4
9	11/12	期中考
10	11/19	Sensor介紹; 通訊模組 Bluetooth, Lora
11	11/26	Sensor介紹; 通訊模組 wifi
12	12/3	Proposal
13	12/10	物聯網平台 - IoT Cloud Platform
14	12/17	AI應用 (SVM)
15	12/24	(期末考周 12/24-30) Project 準備周
16	12/31	(國定假日)
17	1/7	(彈性補充教學) Final demo
18	1/14	(彈性補充教學) Final demo part 2 (如果需要兩周進行)

Last week

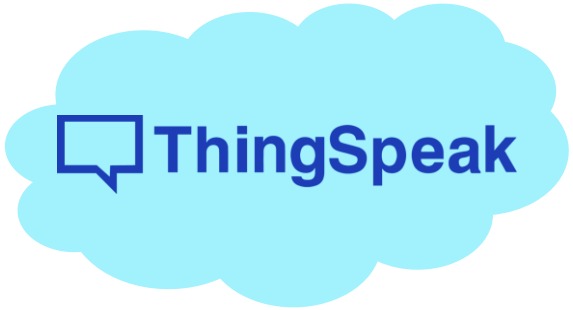
- PIR motion Sensor
  Use HC-SR505 to detect if there is any human
- Sound Sensor
  Use KY-038 to show the sound intensity of surroundings with digital or analog output.
- ThingSpeak APP
  Use TalkBack to enable any device to act upon queued commands.

HC-SR505

PIR Motion Sensor

KY-038

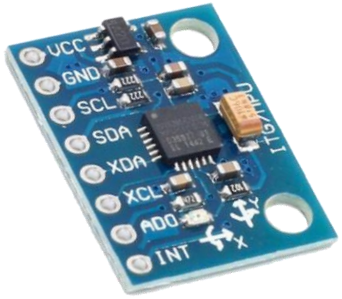
Sound Sensor



This week

- MPU-6050 : Accelerometer + Gyroscope
  - MPU-6000整合了3軸陀螺儀、3軸加速器，可準確追蹤快速與慢速動作。(https://playground.arduino.cc/Main/MPU-6050/)

- Google Colab
  - Google Colab是一個線上的免費虛擬機，透過瀏覽器即可編寫程式，可以連接Google Drive雲端硬碟，儲存訓練完成的AI模型或資料集。(https://colab.research.google.com/)



MPU-6050

Google Colab

Lab. 1. SVM for gesture recognition

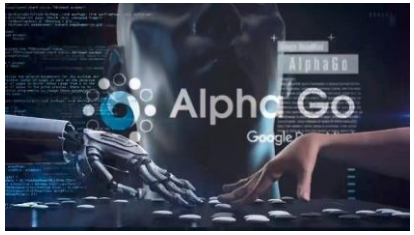
Use Arduino, MPU6050 and SVM to recognition your gestures.

What is AI ?

- AI is an acronym for artificial intelligence. It is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision.

- Famous AI systems:

AlphaGo



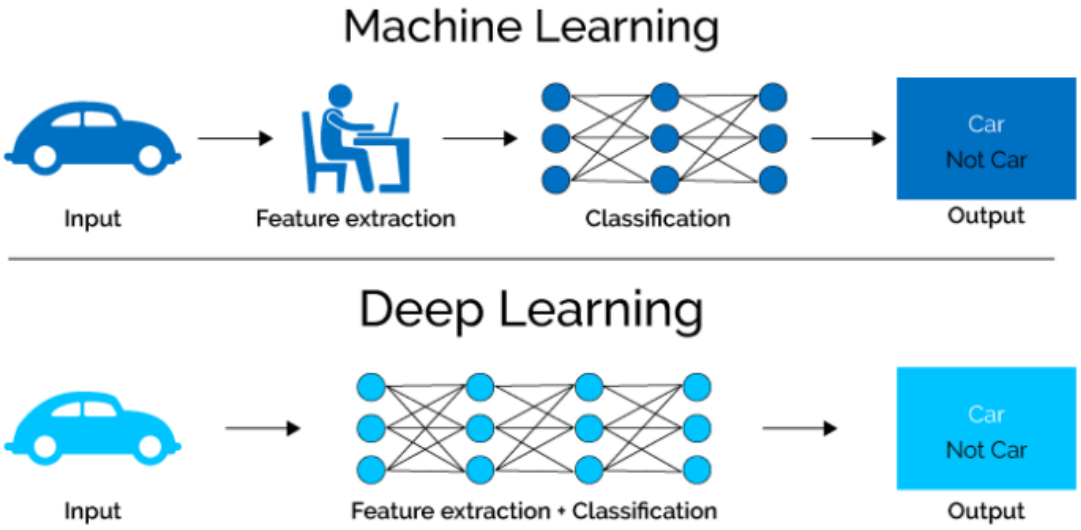
Siri

Sophia

圖片來源
https://unwire.hk/2017/05/22/alphago-vs-ke-jie-match-schedule-released/google/
https://www.newmobilelife.com/2021/10/05/siri-10-years-old/
https://zh-yue.m.wikipedia.org/wiki/File:Sophia_(robot).jpg

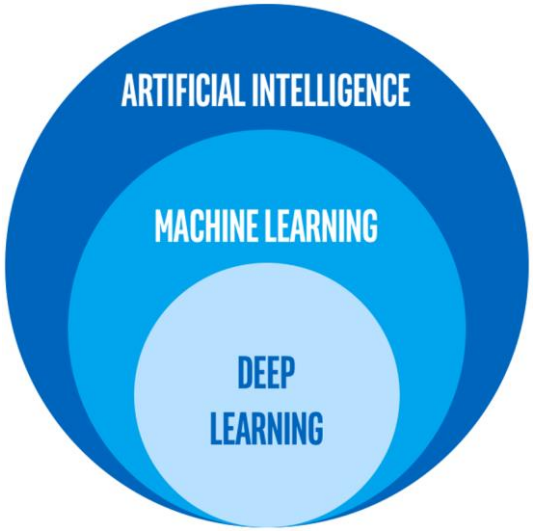
AI (artificial intelligence)

- How to let machines own intelligence?
  - Rule-based programing
  - Machine Learning:
    - Input → Feature Extraction → Model → Answer
  - Deep Learning:
    - Input → Model → Answer



Machine Learning vs. Deep Learning

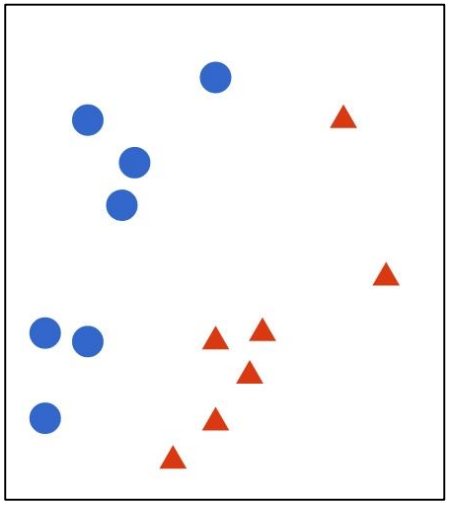
- Deep learning is a type of machine learning, which is a subset of artificial intelligence.
- Machine learning is about computers being able to think and act with less human intervention; deep learning is about computers learning to think using structures modeled on the human brain.
- Machine learning requires less computing power; deep learning typically needs less ongoing human intervention.
- Deep learning can analyze images, videos, and unstructured data in ways machine learning can't easily do.

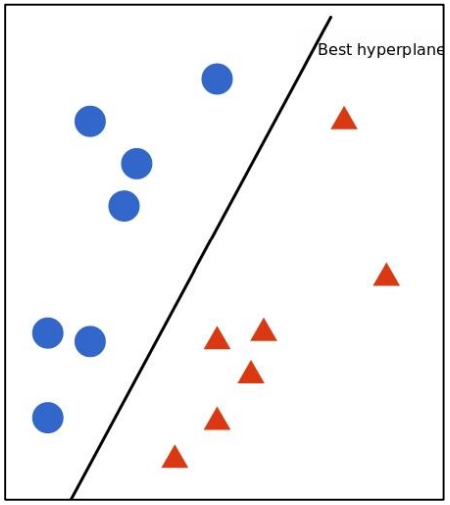


圖片來源
https://www.intel.com/content/www/us/en/artificial-intelligence/posts/difference-between-ai-machine-learning-deep-learning.html

What is SVM ?

- SVM is an acronym for support-vector machine, a famous algorithm in the machine learning.
- SVM is a supervised machine learning model that uses classification algorithms for two-group classification problems.
- The goal of SVM is to find a decision boundary for all clusters.
  - Decision boundary is a hypersurface that partitions the underlying vector space into two sets.





In 2D, the best hyperplane is simply a line

This week

- **Record Gesture**: Use Arduino and MPU6050 to recode gestures and collect the training and testing dataset.
- **Sklearn SVM**: Use scikit-learn library on the Google Colab website to train the gesture recognition model, and output the c code for Arduino.
- **Gesture Recognition**: Use Arduino and MPU6050 to recognition your gestures.

Punch

Figure from:
https://thenounproject.com/term/straight-punch/2893003/

left

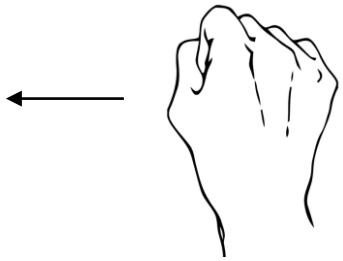


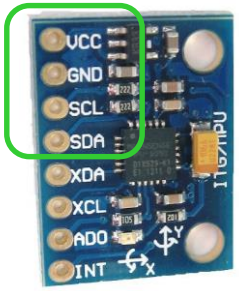
Figure from:
https://www.edupics.com/coloring-page-fist-i16449.html

Lab. 1 Record Gesture

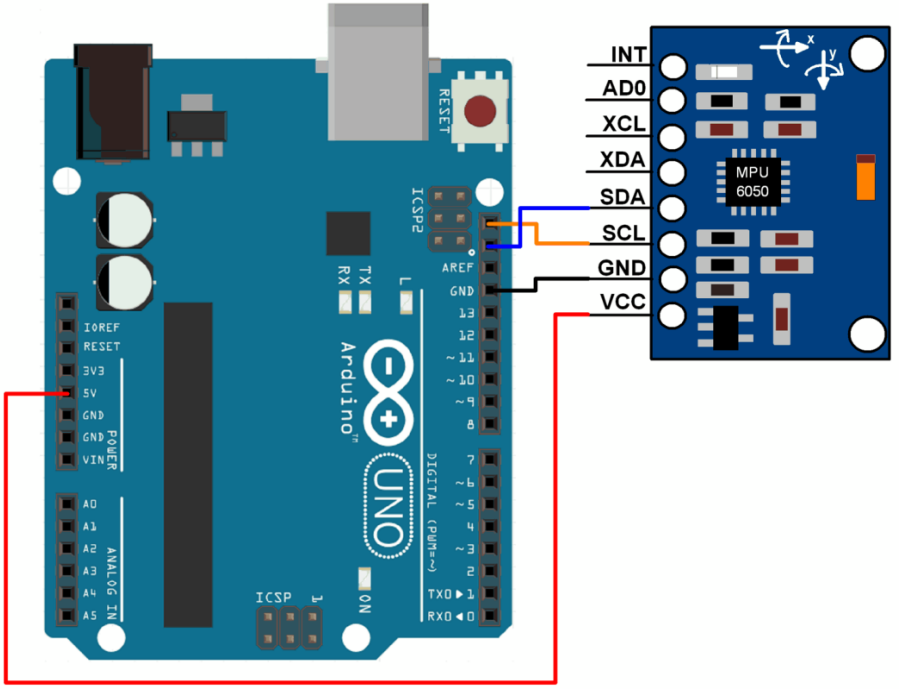
Use Arduino and MPU6050 to recode gestures and collect the training and testing dataset

1. Record gesture

- Goal: Use Arduino and MPU6050 to recode gestures and collect the training and testing dataset.
- Hardware Required
  - Arduino Board
  - MPU-6050



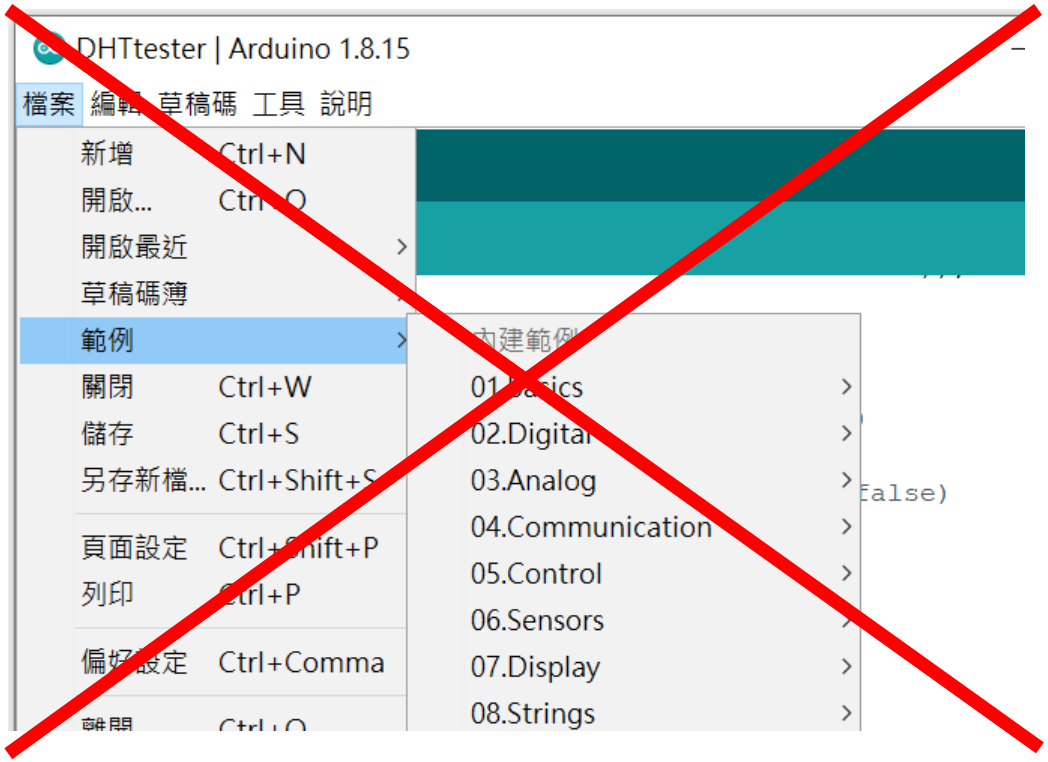
SCL - Serial clock line
SDA - Serial data line



1. Record gesture

Arduino IDE

- Download the library and open the code from E3.
  - Ch.12_ AI application-sample code.zip
- record_gesture → record_gesture.ino



Sample code - record_gesture.ino

宣告與初始

```
4  MPU6050 IMU;
5  //宣告一個MPU6050, 叫作IMU
6  #define NUM_AXES 3
7  //總共加速度有3個方向ax,ay,az(三軸)
8  #define TRUNCATE_AT 20
9  //加速度值的合理範圍, 過濾掉雜訊、不合理的值(spikes)
10 #define ACCEL_THRESHOLD 5
11 //完成一次動作的閥值, 3軸加速度的絕對值總和超過5, 就代表使用者開始在做動作
12 #define NUM_SAMPLES 30
13 //ax,ay,az各方向的測量值數目, 每個方向存30筆資料
14 double baseline[NUM_AXES];
15 //同一瞬間的ax,ay,az的3軸資料
16 float features[NUM_SAMPLES * NUM_AXES];
17 //儲存全部3軸各30筆資料, 共90筆資料(30*3=90)
18
19 void setup() {
20   Serial.begin(115200);
21   //請注意！鮑率是115200！！！
22   while(!IMU.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G))
23   {    Serial.println("Could not find a valid MPU6050 sensor, check wiring!");
24   delay(500);
25   }
26   //初始化MPU6050, 讀取不到會印訊息
27   recordBaseline();
28   //成功初始化後, 進行一次3軸加速度的數值讀取, 作為3軸加速度的基準值
29 }
```

Sample code - record_gesture.ino

初始所用到的函式

```
31  void imu_read(float *ax, float *ay, float *az) {
32    Vector normAccel = IMU.readNormalizeAccel();
33
34    *ax = normAccel.XAxis;
35    *ay = normAccel.YAxis;
36    *az = normAccel.ZAxis;
37  }
38  //此函式負責讀取3軸加速度數值
39
40  void recordBaseline() {
41      float ax, ay, az;
42      for (int i = 0; i < 10; i++) {
43          imu_read(&ax, &ay, &az);
44          delay(100);
45      }
46      baseline[0] = ax;
47      baseline[1] = ay;
48      baseline[2] = az;
49  }
50  //此函式負責讀取3軸加速度數值作為3軸加速度的基準值
```

Sample code - record_gesture.ino

迴圈做的事情

```
88  void loop() {
89      float ax, ay, az;
90      //宣告3個變數儲存3軸加速度值
91      imu_read(&ax, &ay, &az);
92      //讀取&存入3軸加速度值
93      ax = constrain(ax - baseline[0], -TRUNCATE_AT, TRUNCATE_AT);
94      ay = constrain(ay - baseline[1], -TRUNCATE_AT, TRUNCATE_AT);
95      az = constrain(az - baseline[2], -TRUNCATE_AT, TRUNCATE_AT);
96      //判斷ax,ay,az的值, 限制值在正負"TRUNCATE_AT"(此處設定為20)
97      if (!motionDetected(ax, ay, az)) {
98          delay(10);
99          return;
100     }
101     //確定是否成功感測到動作, 沒有就重新偵測
102
103     recordIMU();
104     //讀取30次動作的3軸加速度值
105     printFeatures();
106     //印出30次動作的加速度值
107
108     delay(2000);
109 }
```

- constrain: 限制資料在指定範圍
  EX: 想要把X 限制在 0~255 之間

Sample code - record_gesture.ino

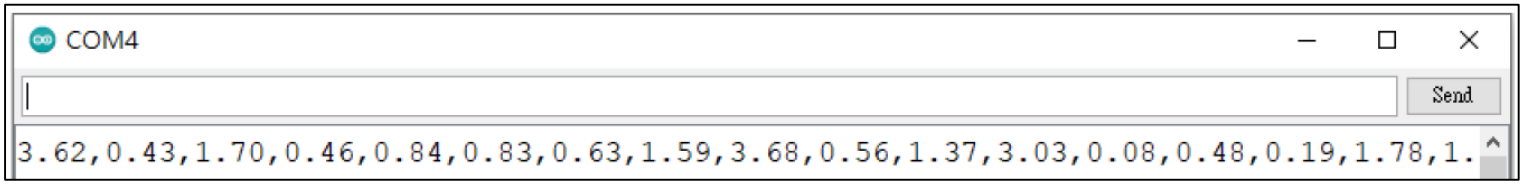
迴圈用到的函式

```
54  bool motionDetected(float ax, float ay, float az) {
55      return (abs(ax) + abs(ay) + abs(az)) > ACCEL_THRESHOLD;
56  }//3軸加速度的絕對值總和超過"ACCEL_THRESHOLD"(此處設定為5)
57  //就代表使用者完成一次動作
58  void recordIMU() {
59      float ax, ay, az;
60      for (int i = 0; i < NUM_SAMPLES; i++) {
61          imu_read(&ax, &ay, &az);
62          ax = constrain(ax - baseline[0], -TRUNCATE_AT, TRUNCATE_AT);
63          ay = constrain(ay - baseline[1], -TRUNCATE_AT, TRUNCATE_AT);
64          az = constrain(az - baseline[2], -TRUNCATE_AT, TRUNCATE_AT);
65          //判斷ax,ay,az的值, 限制值在正負"TRUNCATE_AT"(此處設定為20)
66          features[i * NUM_AXES + 0] = ax;
67          features[i * NUM_AXES + 1] = ay;
68          features[i * NUM_AXES + 2] = az;
69          //將3軸加速度值依序存入陣列
70          delay(30);
71      }
72  }
73  void printFeatures() {
74      const uint16_t numFeatures = sizeof(features) / sizeof(double);
75      //計算"features"陣列的大小(也就是3軸加速度值的總數:90)
76      for (int i = 0; i < numFeatures; i++) {
77          Serial.print(features[i]);
78          Serial.print(i == numFeatures - 1 ? ' ' : ',');
79      }
80      Serial.println(' ');//依序將值印出
81  }
```

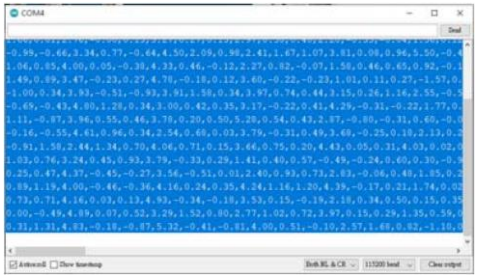
1. Record gesture (3)

- How to create your training dataset and testing dataset

1. Each gesture is represented by the variation of 30 sets of sensing values of ax, ay, az.



2. Do 15 times for a gesture, and then Ctrl a + Ctrl c to record training data.



3. Go to: https://www.convertcsv.com/csv-viewer-editor.htm, and paste results to create a .csv file.

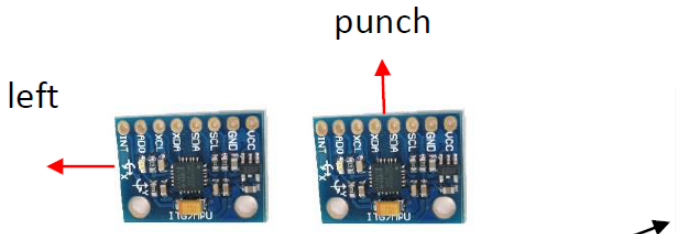
b. Set filename

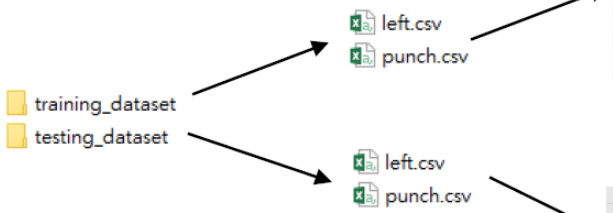
a. click here and ctrl-v

c. download your .csv file

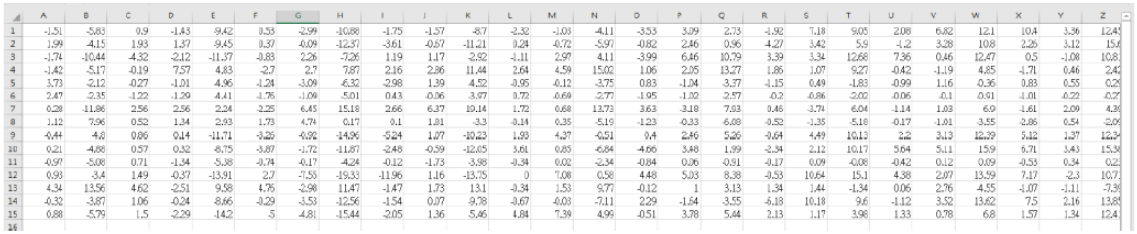
1. Record gesture (4)

- Repeat the process in the previous slide to create your own training dataset and testing dataset for punch and left.
- Record 15 times for each gesture for training and 5 times for each gesture for testing.

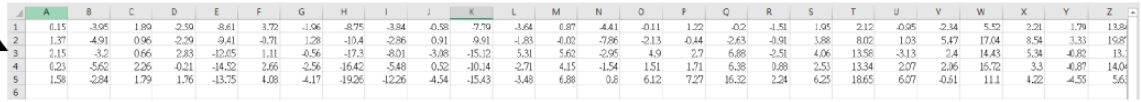




Training data for a gesture



Testing data for a gesture



Sklearn SVM:

Use scikit-learn library on the Google Colab website to train the gesture recognition model, and output the c code for Arduino.

Google COLAB

- Colab is a free Python development environment that runs in the browser using Google Cloud with GPUs.
- TensorFlow, a AI framework developed by Google, is supported in the Colab.

https://colab.research.google.com/notebooks/intro.ipynb

scikit-learn

scikit learn Install User Guide API Examples More Go

scikit-learn

Machine Learning in Python

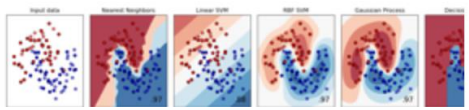
Getting Started | Release Highlights for 1.0 | GitHub

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...



Regression

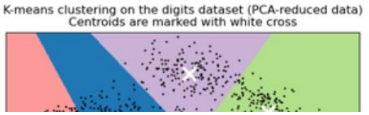
Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

Clustering

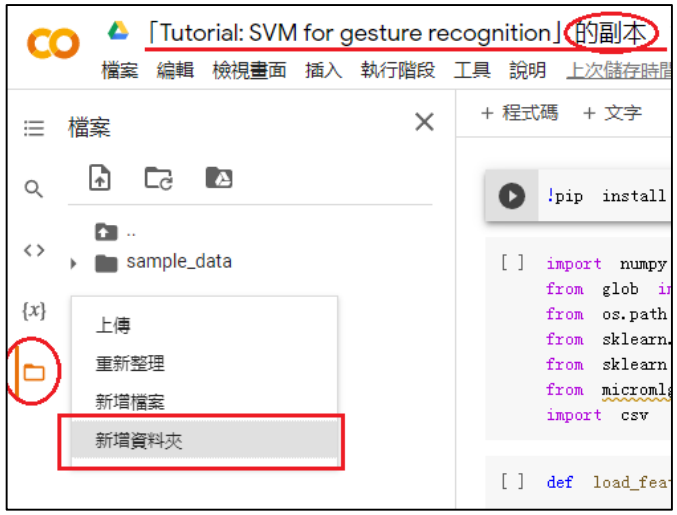
Automatic grouping of similar objects into sets.

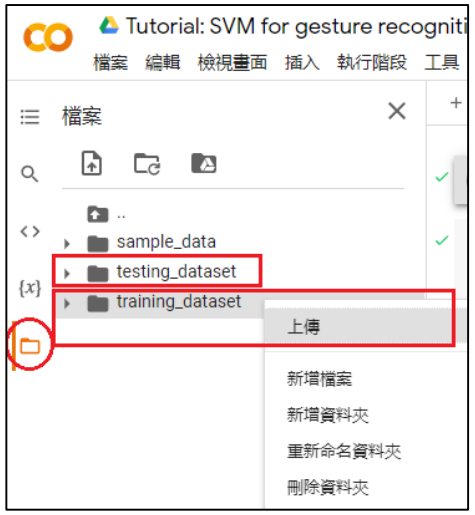
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...



2. Sklearn SVM (1)

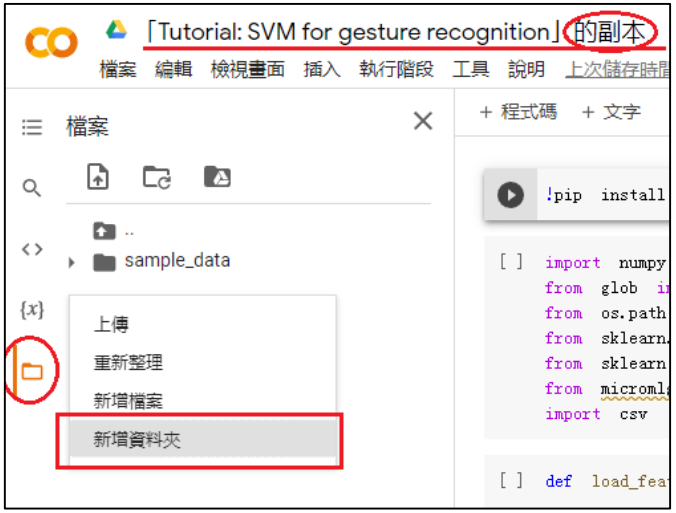
- Go to the Google Colab by following link https://colab.research.google.com/drive/13UeHjuQL_RNcfPXiBJqBn_NmHuCID4gB?usp=sharing
- Create a copy to your own Google cloud.
- Create two folders named training_dataset and testing_dataset, respectively.
- Upload your .csv files to these two folders you just created.

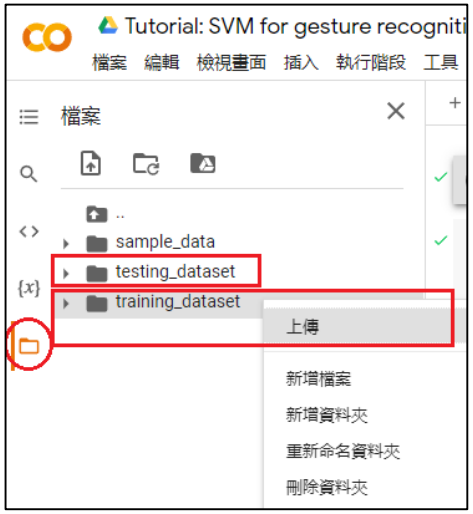




2. Sklearn SVM (1)

- Go to the Google Colab by following link
  https://colab.research.google.com/drive/13UeHjuQL_RNcfPXiBJqBn_NmHuCID4gB?usp=sharing
- Create a copy to your own Google cloud.
- Create two folders named training_dataset and testing_dataset, respectively.
- Upload your .csv files to these two folders you just created.





2. Sklearn SVM (2)

1
```
!pip install micromlgen
```

2
```
[ ] import numpy as np
    from glob import glob
    from os.path import basename
    from sklearn.svm import SVC
    from sklearn import metrics
    from micromlgen import port
    import csv
```

Install and import the libraries we need.

3
```
[ ] def load_features(folder):

        dataset = None
        classmap = {}
        for class_idx, filename in enumerate(glob('%s/*.csv' % folder)):

            class_name = basename(filename)[:-4]
            classmap[class_idx] = class_name
            samples = np.loadtxt(filename, dtype=float,
            labels = np.ones((len(samples), 1)) * class_
            samples = np.hstack((samples, labels))
            dataset = samples if dataset is None else
        return dataset, classmap
```

Load the .csv files from folder to get the training data.

Set parameters of SVM model.

Print the c code for Arduino model.h

4
```
[ ] features, classmap = load_features('training_dataset')
    X_train, y_train = features[:, :-1], features[:, -1]
    classifier = SVC(kernel='poly', gamma=0.002, degree=1).fit(X_train, y_train)
    c_code = port(classifier, classmap=classmap)
    print(c_code)
```

5
```
[ ] features, classmap = load_features('testing_dataset')
    X_test, y_test = features[:, :-1], features[:, -1]
    y_pred = classifier.predict(X_test)
    print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Test the accuracy for your testing set.

2. Sklearn SVM (3)

- You will see the accuracy value for your testing data.

```
features, classmap = load_features('testing_dataset')
X_test, y_test = features[:, :-1], features[:, -1]
y_pred = classifier.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9
```

- Go to Step 4, the results of print(c_code), and then use Ctrl+a and Ctrl+c to **copy this c code model**.

```
features, classmap = load_features('training_dataset')
X_train, y_train = features[:, :-1], features[:, -1]
classifier = SVC(kernel='poly', gamma=0.002, degree=1).fit(X_train, y_train)
c_code = port(classifier, classmap=classmap)
print(c_code)
```

```
#pragma once
#include <cstdarg>
namespace Eloquent {
    namespace ML {
        namespace Port {
            class SVM {
                public:
                    /**
```

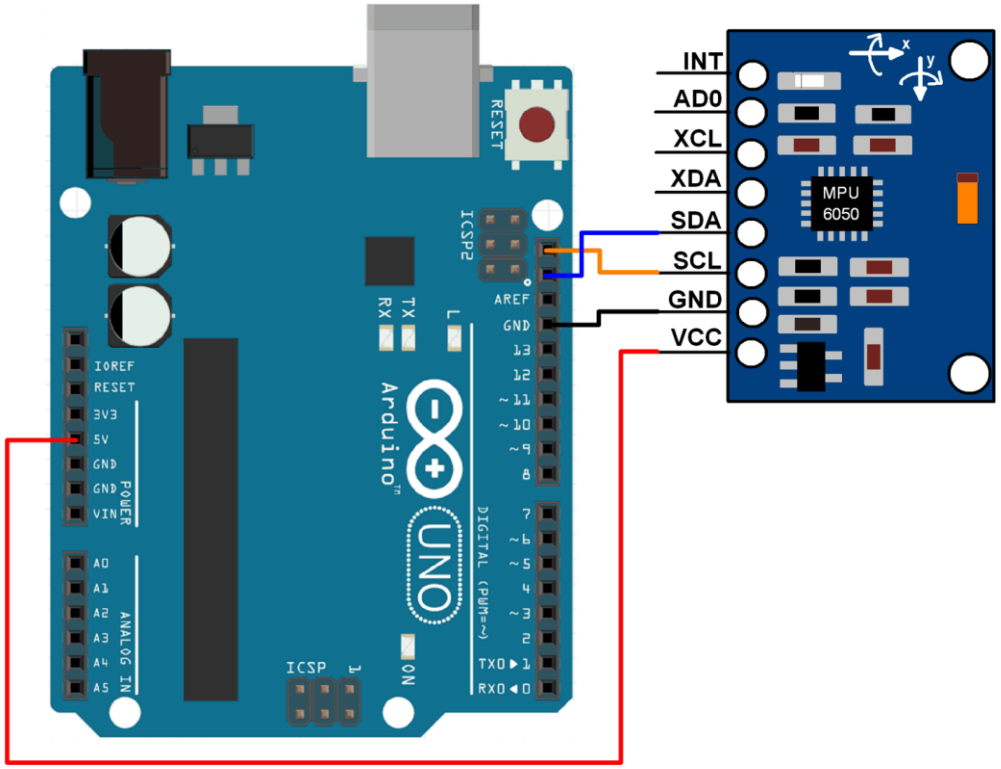
- Ctrl+a
- Ctrl+c

Lab. 3: Gesture Recognition

Use Arduino and MPU6050 to recognition your gestures.

3. Gesture Recognition

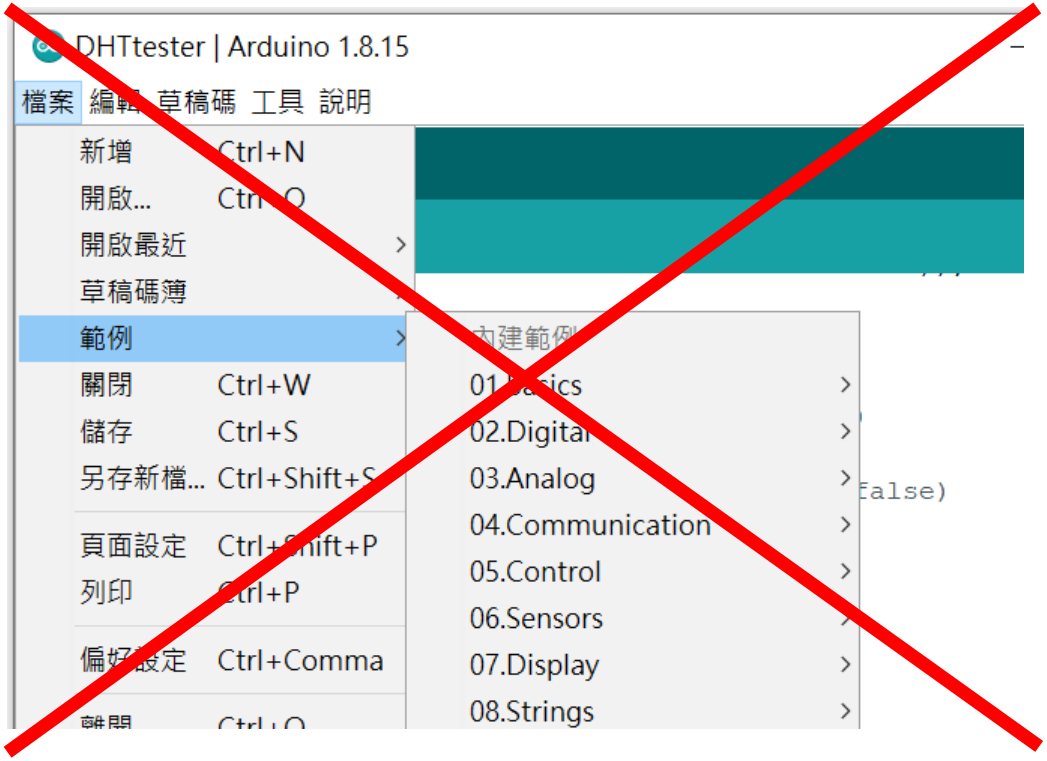
- Goal: Use Arduino and MPU6050 to recognition your gestures.
- Hardware Required
  - Arduino Board
  - MPU-6050



3. Gesture Recognition

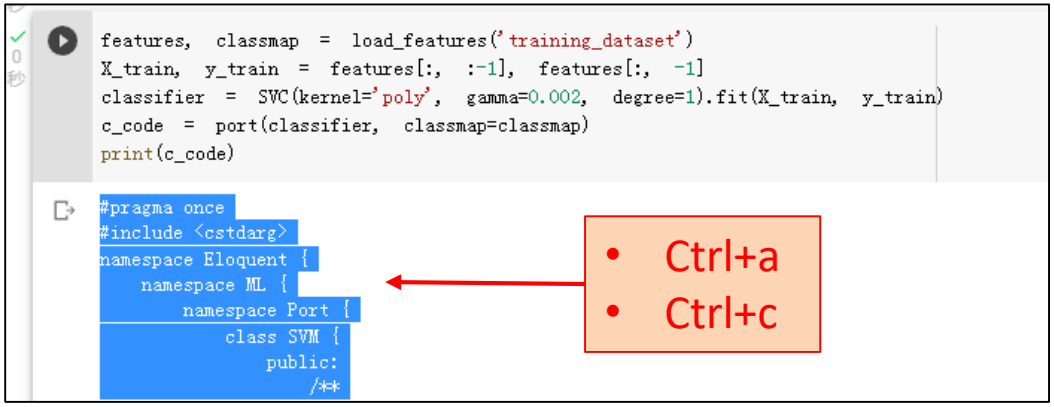
Arduino IDE

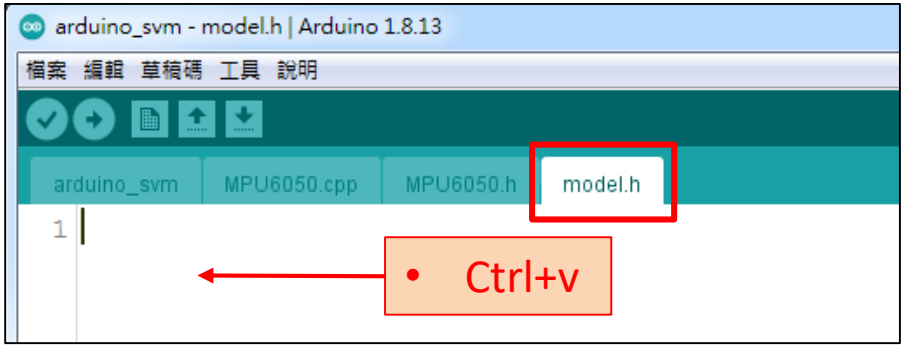
- Download the library and open the code from E3.
  - Ch.12_ AI application-sample code.zip
- **arduino_svm.ino**



3. Gesture Recognition (2)

- Go to the folder of arduino_svm, use Arduino IDE to open the arduino_svm.ino.
- Go to the model.h page to paste the code from Colab.





3. Gesture Recognition (3)

- Comment out the # include <cstdarg>

```
#pragma once
//#include <cstdarg>
namespace Eloquent {
    namespace ML {
        namespace Port {
```

- Upload the arduino_svm to your Arduino and test your gestures

Sample code - arduino_svm.ino

- The arduino_svm.ino is almost the same as record_gesture.ino, so we only introduce the difference between them.

arduino_svm.ino

```
void loop() {
  // display the data
    float ax, ay, az;

    imu_read(&ax, &ay, &az);
    ax = constrain(ax - baseline[0], -TRUNCATE_A
    ay = constrain(ay - baseline[1], -TRUNCATE_A
    az = constrain(az - baseline[2], -TRUNCATE_A
    if (!motionDetected(ax, ay, az)) {
        delay(10);
        return;
    }

    recordIMU();
    classify();
    delay(2000);
}

void classify() {
    Serial.print("Detected gesture: ");
    Serial.println(clf.predictLabel(features));
}
```

model.h

```
/**
* Convert class idx to readable
*/
const char* predictLabel(float *x) {
    switch (predict(x)) {
        case 0:
        return "punch";
        case 1:
        return "left";
        default:
        return "Houston we have a problem";
    }
}
```

Sample code - arduino_svm.ino

- According to your input feature values to do a math with kernels,
  - if the decision > 0, the gesture is punch;
  - else if (the decision < 0), the gesture is left.

model.h

```
int predict(float *x) {
    float kernels[18] = { 0 };
    float decisions[1] = { 0 };
    int votes[2] = { 0 };
    kernels[0] = compute_kernel(x,   -4.26  , 1.71  , 1.28  , -8.9  , 1.0
    kernels[1] = compute_kernel(x,   -3.97  , 1.99  , 0.85  , -8.72  , 1.
    kernels[2] = compute_kernel(x,   -5.13  , 2.37  , 1.91  , -9.48  , 2.
    kernels[3] = compute_kernel(x,   3.19  , -2.74  , -2.8  , 1.8  , -0.4
    kernels[4] = compute_kernel(x,   17.87  , 2.58  , -5.92  , 14.83  , 5
    kernels[5] = compute_kernel(x,   18.39  , -4.9  , 0.54  , 18.39  , 3.
    kernels[6] = compute_kernel(x,   3.33  , -8.24  , -8.96  , 18.39  , -
    kernels[7] = compute_kernel(x,   11.37  , -4.17  , -5.5  , 9.08  , -6
    kernels[8] = compute_kernel(x,   -6.67  , -0.21  , -0.55  , -4.11  ,
    kernels[9] = compute_kernel(x,   -10.14  , 2.96  , 3.99  , -11.99  ,
    kernels[10] = compute_kernel(x,   -1.51  , -5.83  , 0.9  , -1.43  , -
    kernels[11] = compute_kernel(x,   1.99  , -4.15  , 1.93  , 1.37  , -9
    kernels[12] = compute_kernel(x,   -1.42  , -5.17  , -0.19  , 7.57  ,
    kernels[13] = compute_kernel(x,   3.73  , -2.12  , -0.27  , -1.01  ,
    kernels[14] = compute_kernel(x,   2.47  , -2.35  , -1.22  , -1.29  ,
    kernels[15] = compute_kernel(x,   1.12  , 7.96  , 0.52  , 1.34  , 2.9
    kernels[16] = compute_kernel(x,   -0.97  , -5.08  , 0.71  , -1.34  ,
    kernels[17] = compute_kernel(x,   4.34  , 13.56  , 4.62  , -2.51  , 9
    float decision = 0.823207791772;
    decision = decision - ( + kernels[0] * -0.109131545125  + kernels[1]
    decision = decision - ( + kernels[10] * 0.474839242088  + kernels[11]

    return decision > 0 ? 0 : 1;
}
```

model.h

```
/**
* Convert class idx to readal
*/
const char* predictLabel(float *x) {
    switch (predict(x)) {
        case 0:
        return "punch";
        case 1:
        return "left";
        default:
        return "Houston we have a problem";
    }
}
```

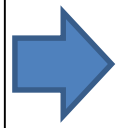
Fail to recognize gestures?

- If you have trouble to recognize your gestures, you may add the Serial.println (decision) before returning the decision in the model.h as below:

```
Serial.println(decision);
return decision > 0 ? 0 : 1;
```

- And you can observe the decision boundary between two gestures, and determine a suitable value to change decision boundary.

```
Serial.println(decision);
return decision > 0 ? 0 : 1;
```



```
Serial.println(decision);
return decision > 1 ? 0 : 1;
```

Quiz 1

- Add a new gesture and choose one gesture introduced in the tutorials to recognize these two gestures by using Arduino.
  - Example : **Up** and left

Summary

- **“請記得填寫”教室座位實聯制**
  - https://docs.google.com/spreadsheets/d/1k4q-JP9Pk9cLGY70V04Nbc6XbUbBdYu_TXqJtHF6rGk
- **Practice Labs by yourself**
- **Write Answers for Discussion**
  - No discussion this week
- **Quiz: Write code for quiz, then demonstrate to TAs**
  - Quiz 1 . Add a new gesture