機器學習與實作
需求+分析

0652016許銘哲

0652075黃子衡

需求及目的

- 接到95%的球
  1. Rule Base為主
  2. 以Rule Base產生記錄檔輔助機器學習(P1為SVM，P2為KNN)
  3. 能化解奇怪的球形
- 接到球的位置是在板子中央正負5處
  1. 須提前預知球的位置(導入馬可夫鏈輔助)
- 攻擊：能改變球的速度與方向(切球)
  1. 接到球的瞬間左右搖晃
  2. 攻擊方式待試

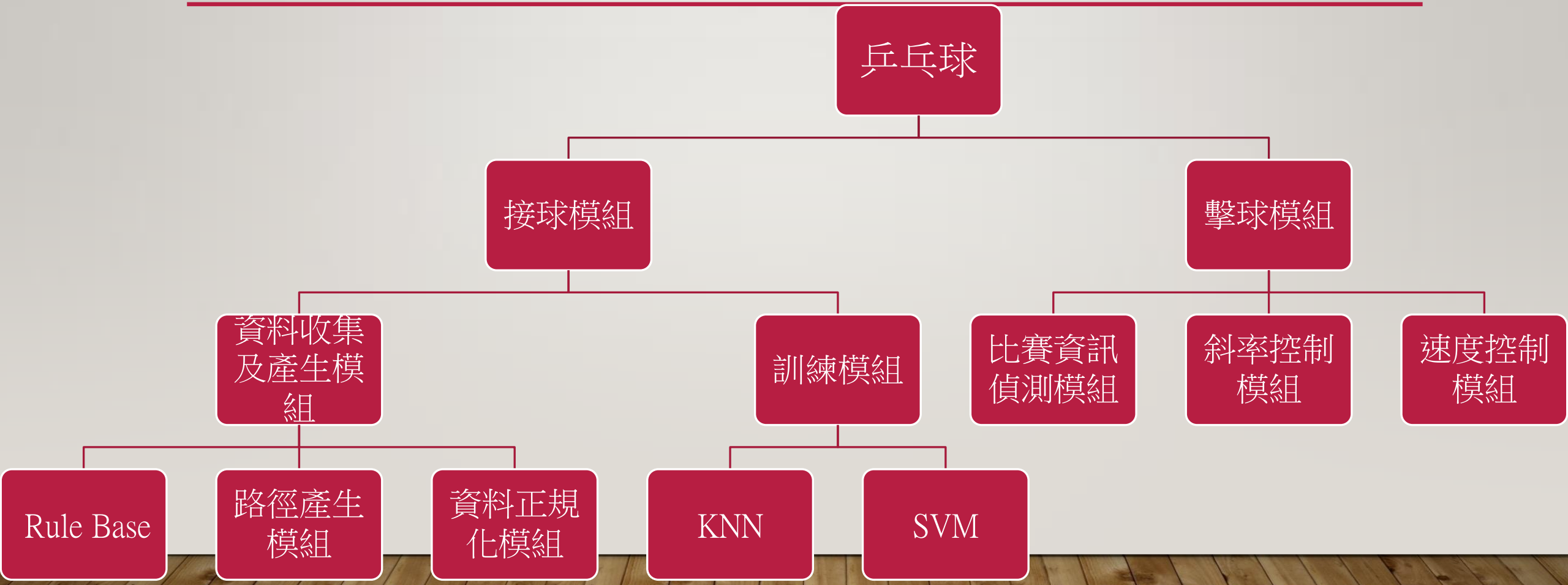
分析 接到95%的球

- 資料分析(產生大量及有效的樣本)
  1. Rule Base產生一部分樣本
     ① 以亂數方式決定發球的位置及方向
  2. 產生球的路徑(可視化)
     ① 擷取接球位置及碰撞點(求斜率)
     ② 接到球時標計碰撞點並推導出其方程式或遊戲內路徑
  3. 更新非已知資料
     ① 用人工找出路徑資料的空缺點並修正

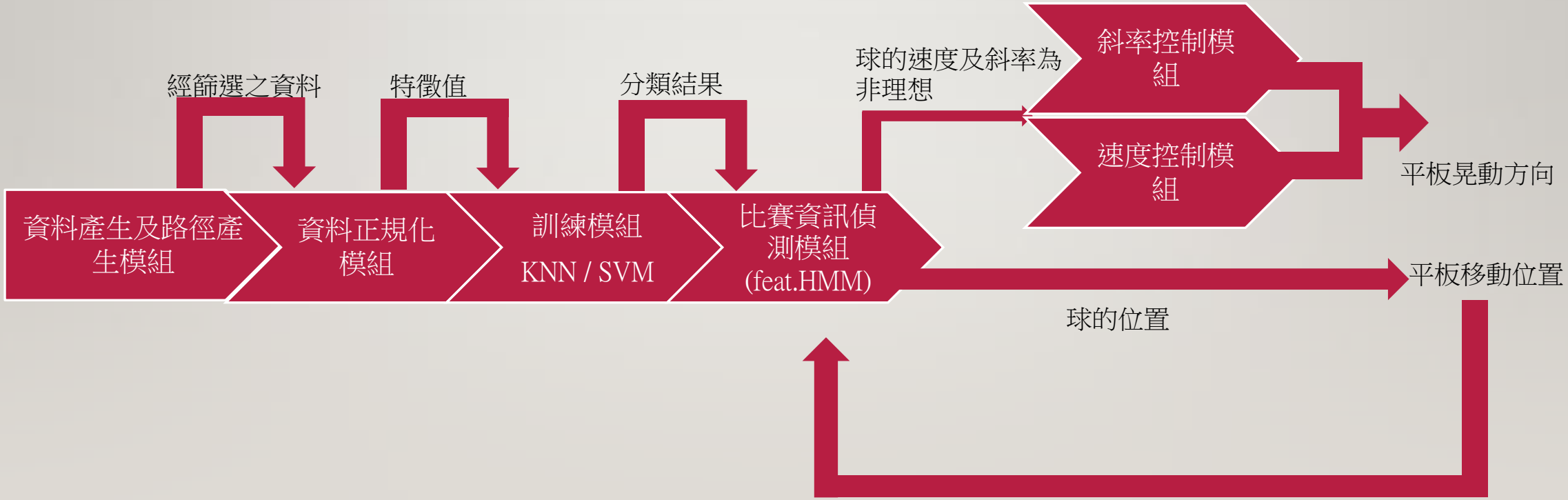
分析 接到95%的球

- 接球模式
  1. 將過快的球(球速為12)減速
     ① 逆著球的方向切球
  2. 將角度過大的球(斜率大於50度或斜率小於40度)修正至50度~40度之間
     ① 斜率過大時順著球的方向切球
     ② 斜率過小時逆著球的方向切球

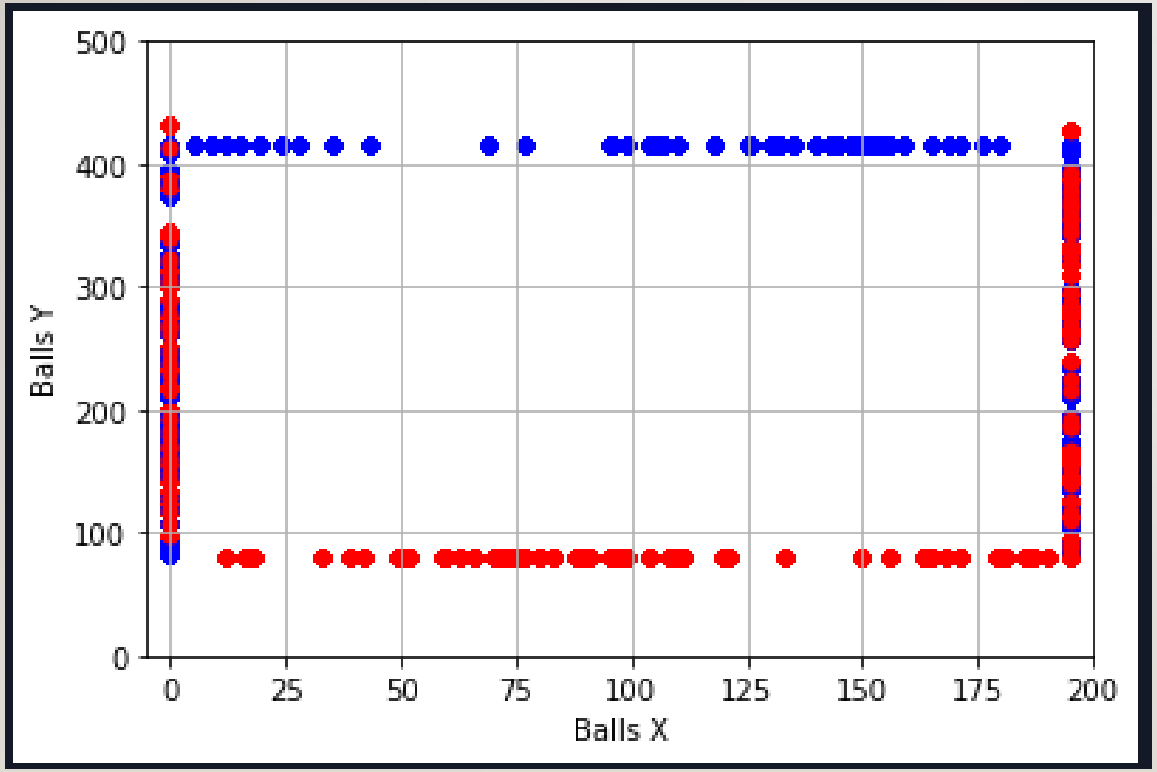
分析簡圖


乒乓球
接球模組
擊球模組
資料收集及產生模組
訓練模組
比賽資訊偵測模組
斜率控制模組
速度控制模組
Rule Base
路徑產生模組
資料正規化模組
KNN
SVM

架構圖


經篩選之資料
特徵值
分類結果
球的速度及斜率為非理想
斜率控制模組
速度控制模組
平板晃動方向
資料產生及路徑產生模組
資料正規化模組
訓練模組
KNN / SVM
比賽資訊偵測模組
(feat.HMM)
平板移動位置
球的位置

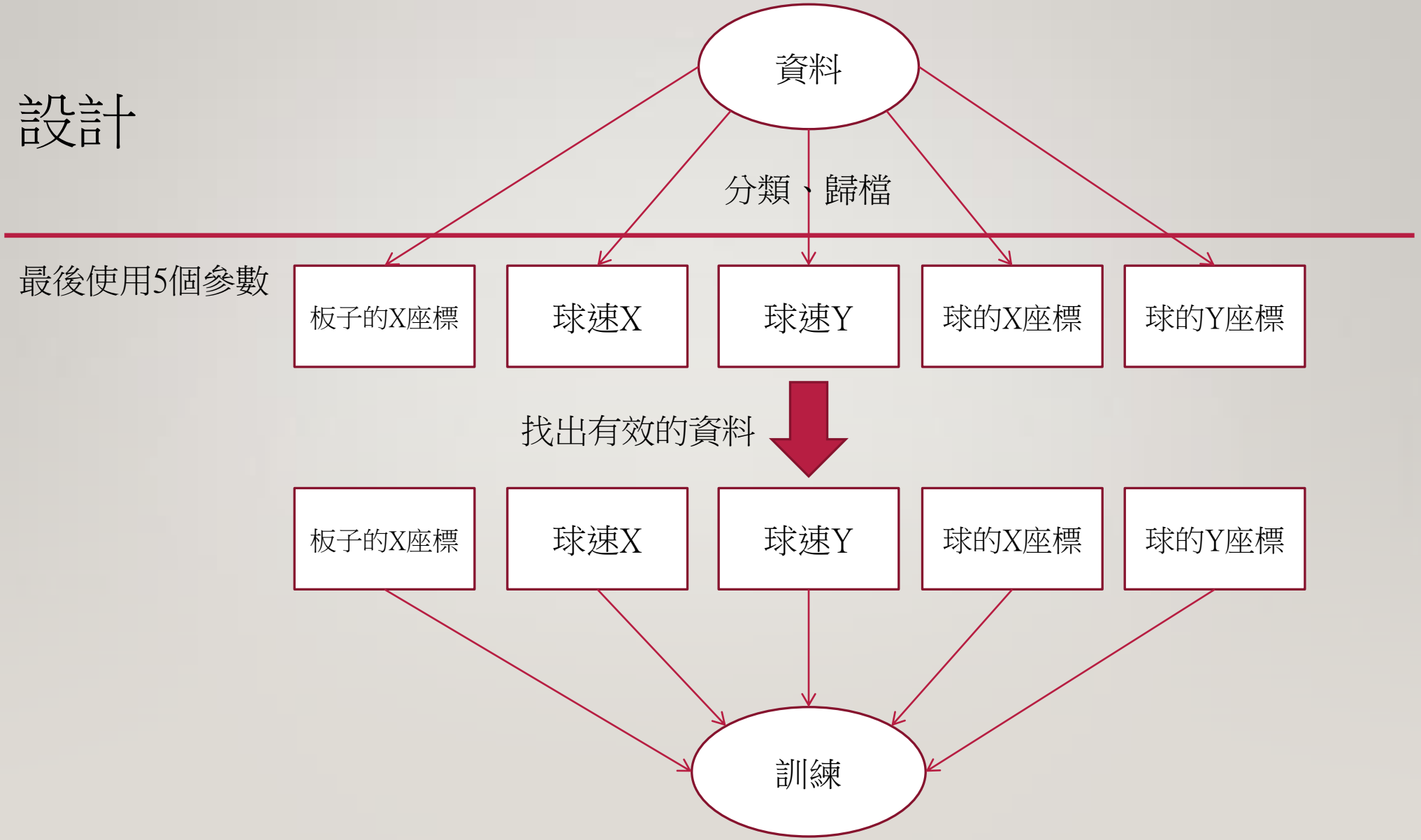
資料可視化



只取球的座標在Y在415和80，X在0和195，並將他們顯示出來，是否可成為參考資料。

球向下為紅色，球向上為藍色。

設計


資料
分類、歸檔
最後使用5個參數
板子的X座標
球速X
球速Y
球的X座標
球的Y座標
找出有效的資料
板子的X座標
球速X
球速Y
球的X座標
球的Y座標
訓練

程式(訓練KNN)

KNN_2Player.py | knn_1p.py

```python
for f in files:         ##將路徑底下的檔名與路徑結合
    allpath = join(path, f)
    if isfile(allpath):
        with open(allpath , "rb") as fl:
            data_list1 = pickle.load(fl)
            for ml_name in data_list1.keys():
                if ml_name == "record_format_version":
                    continue
            target_record = data_list1[ml_name]
            for n in range(0,len(target_record["scene_info"])):
                Frame.append(target_record["scene_info"][n]["frame"])
                PlatformPosition1P.append(target_record["scene_info"][n]["platform_1P"])
                BallPosition.append(target_record["scene_info"][n]["ball"])
                BallSpeed.append(target_record["scene_info"][n]["ball_speed"])

PlatX = np.array(PlatformPosition1P) [:,0][:,np.newaxis] #[:,0]->取所有第一陣列的第一個數值(X座標) #[:,np.newaxis]->陣列變成直的
PlatX_next = PlatX[1:,:] #除了第一個值以外都要
instrust = (PlatX_next-PlatX[0:len(PlatX_next)])/5 #板子位移量為5

Ballarray = np.array(BallPosition[:-1])

Ball_Vx=np.array(BallSpeed)[:-1,0][:,np.newaxis]

Ball_Vy=np.array(BallSpeed)[:-1,1][:,np.newaxis]
x = np.hstack((Ballarray,PlatX[:-1,0][:,np.newaxis],Ball_Vx,Ball_Vy))

np.set_printoptions(threshold=np.inf)
y = instrust
y = np.array(y, dtype=int)


np.set_printoptions(threshold=np.inf)
#--------------------------- train & test data
from sklearn.model_selection import train_test_split

#--------------------------- train model
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score ,mean_squared_error
from sklearn.feature_selection import SelectKBest, f_regression,GenericUnivariateSelect



x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 1)
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(x_train,y_train)

yknn_bef_scaler = knn.predict(x_test)
r2 = accuracy_score(yknn_bef_scaler,y_test)
mse = mean_squared_error(y_test, yknn_bef_scaler)

filename = "C:\\Users\\aiolb\\Desktop\\MLGame-beta8.0.1\\games\\pingpong\\ml\\knn_1P_20210106.sav"
pickle.dump(knn,open(filename,"wb"))
```

KNN_2Player.py | knn_1p.py | knn_2P.py

```python
for f in files:         ##將路徑底下的檔名與路徑結合
    allpath = join(path, f)
    if isfile(allpath):
        with open(allpath , "rb") as fl:
            data_list1 = pickle.load(fl)
            for ml_name in data_list1.keys():
                if ml_name == "record_format_version":
                    continue
            target_record = data_list1[ml_name]
            for n in range(0,len(target_record["scene_info"])):
                Frame.append(target_record["scene_info"][n]["frame"])
                PlatformPosition2P.append(target_record["scene_info"][n]["platform_2P"])
                BallPosition.append(target_record["scene_info"][n]["ball"])
                BallSpeed.append(target_record["scene_info"][n]["ball_speed"])

PlatX = np.array(PlatformPosition1P) [:,0][:,np.newaxis] #[:,0]->取所有第一陣列的第一個數值(X座標) #[:,np.newaxis]->陣列變成直的
PlatX_next = PlatX[1:,:] #除了第一個值以外都要
instrust = (PlatX_next-PlatX[0:len(PlatX_next)])/5 #板子位移量為5

Ballarray = np.array(BallPosition[:-1])

Ball_Vx=np.array(BallSpeed)[:-1,0][:,np.newaxis]

Ball_Vy=np.array(BallSpeed)[:-1,1][:,np.newaxis]
x = np.hstack((Ballarray,PlatX[:-1,0][:,np.newaxis],Ball_Vx,Ball_Vy))

np.set_printoptions(threshold=np.inf)
y = instrust
y = np.array(y, dtype=int)


np.set_printoptions(threshold=np.inf)
#--------------------------- train & test data
from sklearn.model_selection import train_test_split

#--------------------------- train model
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score ,mean_squared_error
from sklearn.feature_selection import SelectKBest, f_regression,GenericUnivariateSelect



x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 1)
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(x_train,y_train)

yknn_bef_scaler = knn.predict(x_test)
r2 = accuracy_score(yknn_bef_scaler,y_test)
mse = mean_squared_error(y_test, yknn_bef_scaler)

filename = "C:\\Users\\aiolb\\Desktop\\MLGame-beta8.0.1\\games\\pingpong\\ml\\knn_2P_20210106.sav"
pickle.dump(knn,open(filename,"wb"))
```

程式(訓練SVM)

KNN_2Player.py | knn_1p.py | knn_2P.py | svm_1p.py | svm_2p.py

```python
for f in files:         ##將路徑底下的檔名與路徑結合
    allpath = join(path, f)
    if isfile(allpath):
        with open(allpath , "rb") as f1:
            data_list1 = pickle.load(f1)
            for ml_name in data_list1.keys():
                if ml_name == "record_format_version":
                    continue
            target_record = data_list1[ml_name]
            for n in range(0,len(target_record["scene_info"])):
                Frame.append(target_record["scene_info"][n]["frame"])
                PlatformPosition1P.append(target_record["scene_info"][n]["platform_1P"])
                BallPosition.append(target_record["scene_info"][n]["ball"])
                BallSpeed.append(target_record["scene_info"][n]["ball_speed"])


from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score ,mean_squared_error


PlatX = np.array(PlatformPosition2P) [:,0][:,np.newaxis] #[:,0]->取所有第一陣列的第一個數值(X座標)
PlatX_next = PlatX[1:,:] #除了第一個值以外都要
instrust = (PlatX_next-PlatX[0:len(PlatX_next)])/5 #板子位移量為5

Ballarray = np.array(BallPosition[:-1])
Ball_Vx=np.array(BallSpeed)[:-1,0][:,np.newaxis]

Ball_Vy=np.array(BallSpeed)[:-1,1][:,np.newaxis]
x = np.hstack((Ballarray,PlatX[:-1,0][:,np.newaxis],Ball_Vx,Ball_Vy))
          #球座標X,球座標Y        板子X              球速X,球速Y

np.set_printoptions(threshold=np.inf)
y = instrust
y = np.array(y, dtype=int)
print(len(instrust))

svr = SVR(gamma=0.001,C = 1,epsilon = 0.1,kernel = 'rbf')

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 40)

svr.fit(x_train,y_train)
y_predict = svr.predict(x_test)
filename = "C:\\Users\\aiolb\\Desktop\\MLGame-beta8.0.1\\games\\pingpong\\ml\\SVM_1P_20210106.sav"
pickle.dump(svr,open(filename,"wb"))
```

KNN_2Player.py | knn_1p.py | knn_2P.py | svm_1p.py | svm_2p.py

```python

for f in files:         ##將路徑底下的檔名與路徑結合
    allpath = join(path, f)
    if isfile(allpath):
        with open(allpath , "rb") as f1:
            data_list1 = pickle.load(f1)
            for ml_name in data_list1.keys():
                if ml_name == "record_format_version":
                    continue
            target_record = data_list1[ml_name]
            for n in range(0,len(target_record["scene_info"])):
                Frame.append(target_record["scene_info"][n]["frame"])
                PlatformPosition2P.append(target_record["scene_info"][n]["platform_2P"])
                BallPosition.append(target_record["scene_info"][n]["ball"])
                BallSpeed.append(target_record["scene_info"][n]["ball_speed"])


from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score ,mean_squared_error


PlatX = np.array(PlatformPosition2P) [:,0][:,np.newaxis] #[:,0]->取所有第一陣列的第一個數值(X座標)
PlatX_next = PlatX[1:,:] #除了第一個值以外都要
instrust = (PlatX_next-PlatX[0:len(PlatX_next)])/5 #板子位移量為5

Ballarray = np.array(BallPosition[:-1])
Ball_Vx=np.array(BallSpeed)[:-1,0][:,np.newaxis]

Ball_Vy=np.array(BallSpeed)[:-1,1][:,np.newaxis]
x = np.hstack((Ballarray,PlatX[:-1,0][:,np.newaxis],Ball_Vx,Ball_Vy))
          #球座標X,球座標Y        板子X              球速X,球速Y

np.set_printoptions(threshold=np.inf)
y = instrust
y = np.array(y, dtype=int)
print(len(instrust))

svr = SVR(gamma=0.001,C = 1,epsilon = 0.1,kernel = 'rbf')

svr.fit(x_train,y_train)
y_predict = svr.predict(x_test)

filename = "C:\\Users\\aiolb\\Desktop\\MLGame-beta8.0.1\\games\\pingpong\\ml\\SVM_2P_20210106.sav"
pickle.dump(svr,open(filename,"wb"))
```

程式(遊玩)

KNN_1Player.py

```
23
24      global wait_frame
25      while True:
26          if scene_info["status"] != "GAME_ALIVE":     #比出勝負
27              return "RESET"  #遊戲重製
28          if not self.ball_served:     #如果未發球
29              self.ball_served = True
30              #print("ball_pos:",scene_info["ball"])
31              if(ball_served_random==1):
32                  return "SERVE_TO_RIGHT"  #往右發球
33              else:
34                  return "SERVE_TO_LEFT"  #往左發球
35
36          else:
37              ball_position_history.append(scene_info["ball"])
38              BallPosition=np.asarray(ball_position_history[-1])
39              PlatX = np.asarray(scene_info["platform_1P"][-2])
40              Ball_Vx=np.asarray(scene_info["ball_speed"][-2])
41              Ball_Vy=np.asarray(scene_info["ball_speed"][-1])
42              data_x = np.hstack((BallPosition,PlatX,Ball_Vx,Ball_Vy))
43              input_data_x = data_x[np.newaxis, :]
44              move = model.predict(input_data_x)
45              print("1P=",move)
46
47              if(move <0):
48                  return "MOVE_LEFT"
49              elif(move >0):
50                  return "MOVE_RIGHT"
51              else:
52                  return "NONE"
53              return "NONE"
54
55
56
57      def reset(self):
58          """
59          Reset the status
60          """
61          global ball_served_random
62          self.ball_served = False
63          ball_served_random=random.randrange(1,3)
64
```

KNN_2Player.py

```
23
24      global wait_frame
25      while True:
26          if scene_info["status"] != "GAME_ALIVE":     #比出勝負
27              return "RESET"  #遊戲重製
28          if not self.ball_served:     #如果未發球
29              self.ball_served = True
30              #print("ball_pos:",scene_info["ball"])
31              if(ball_served_random==1):
32                  return "SERVE_TO_RIGHT"  #往右發球
33              else:
34                  return "SERVE_TO_LEFT"  #往左發球
35
36          else:
37              ball_position_history.append(scene_info["ball"])
38              BallPosition=np.asarray(ball_position_history[-1])
39              PlatX = np.asarray(scene_info["platform_2P"][-2])
40              Ball_Vx=np.asarray(scene_info["ball_speed"][-2])
41              Ball_Vy=np.asarray(scene_info["ball_speed"][-1])
42              data_x = np.hstack((BallPosition,PlatX,Ball_Vx,Ball_Vy))
43              input_data_x = data_x[np.newaxis, :]
44              move = model.predict(input_data_x)
45              print("2P=",move)
46
47              if(move <0):
48                  return "MOVE_LEFT"
49              elif(move>0):
50                  return "MOVE_RIGHT"
51              else:
52                  return "NONE"
53              return "NONE"
54
55
56
57      def reset(self):
58          """
59          Reset the status
60          """
61          global ball_served_random
62          self.ball_served = False
63          ball_served_random=random.randrange(1,3)
64
```

驗證

初賽:

```
1P: 0652075, 0652016
2P: 0652068, 0652001, 0652013
比賽結果：
EASY: 0:3 2P win
第一回: 速度21 2P win
第二回: 速度21 2P win
第三回: 速度23 2P win
-
NORMAL: 0:3 2P win
第一回: 速度21 2P win
第二回: 速度21 2P win
第三回: 速度21 2P win
-
HARD: 3:2 1P win
第一回: 速度7 2P win
第二回: 速度8 1P win
第三回: 速度12 1P win
第四回: 速度14 2P win
第五回: 速度14 1P win
```

複賽:

```
1P: C3 組員: 0652050
2P: A3 組員: 0652075 0652016

比賽結果：
EASY:  0:3 2P WIN
第一回合 : 速度 22 2P WIN
第二回合 : 速度 15 2P WIN
第三回合 : 速度 22 2P WIN

NORMAL:  0:3 2P WIN
第一回合 : 速度 12 2P WIN
第二回合 : 速度 13 2P WIN
第三回合 : 速度 12 2P WIN

HARD:  2:3  2P WIN
第一回合 : 速度 13 1P WIN
第二回合 : 速度 8 2P WIN
第三回合 : 速度 11 1P WIN
第四回合 : 速度 10 2P WIN
第五回合 : 速度 15 2P WIN
```

```
1P: A3 組員: 0652075 0652016
2P: C4 組員: 109112128  109112118

比賽結果：
EASY:  0:3 2P WIN
第一回合 : 速度20 2P WIN
第二回合 : 速度25 2P WIN
第三回合 :  速度20 2P WIN

NORMAL: 3:0 1P WIN
第一回合 : 速度19 1P WIN
第二回合 : 速度21 1P WIN
第三回合 : 速度19 1P WIN

HARD:  1:3  2P WIN
第一回合 : 速度15 1P WIN
第二回合 : 速度16 2P WIN
第三回合 : 速度10 2P WIN
第四回合 : 速度20 2P WIN
```