

# 基於遊戲的機器學習入門

## Final 報告

張君豪

# 基本想法、做法 -- rule

- 一開始，打磚塊時，依反彈程度(照對稱三角形看球會到板前或後)，來做移動。

Handwritten notes and diagrams illustrating the logic for the brick game.

**Diagrams:**

- Top left: A coordinate system showing a ball at  $(a, b)$  before and after a bounce. The distance from the ball to the paddle is  $L$ .
- Middle left: A diagram showing a ball at  $(x+2.5, y+5)$  with a distance of 50 to the paddle.
- Bottom left: A diagram showing a ball at  $(x+2.5, y+5)$  with a distance of 50 to the paddle.
- Bottom right: A diagram showing a ball at  $(x+2.5, y+5)$  with a distance of 50 to the paddle.

**Formulas:**

- $(d-b) > 0$  and  $c-a > 0$
- $l = 200 - c$
- $L = (200 - c) \times \frac{d-b}{c-a}$

**Logic Flow:**

- if  $2L < (400 - d)$   
板子往左
- elif  $2L > (400 - d)$   
板子向右
- if  $2L > 400 - d - 5$   
and  
 $2L < 400 - d + 5$   
板子不動

**take note:**  
Python 不支援 縮排時 Space 和 Tab 混用

**Final Formula:**

- $l = c - d = c$
- $L = c \times \frac{d-b}{a-c}$

**Additional Diagrams:**

- Bottom left: A diagram showing a ball at  $(x+2.5, y+5)$  with a distance of 50 to the paddle.
- Bottom right: A diagram showing a ball at  $(x+2.5, y+5)$  with a distance of 50 to the paddle.

 ml\_play\_original\_rule\_pass\_pass\_game1.py 

```

69 → → → → → #.3.4. Send the instruction for this frame to the game process ..... CR LF
70 → → → → → #if scene_info.frame>1300 and scene_info.frame<1500: CR LF
71 → → → → → # → comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CR LF
72 → → → → → # → instruct.append(-1) CR LF
73 → → → → → if (400-d)<100: CR LF
74 → → → → → if (c<80): CR LF
75 → → → → → if (d-b)>0 and (c-a)<0: CR LF
76 → → → → → L=c*(d-b)/(a-c) CR LF
77 → → → → → sL=abs(400-d-L) CR LF
78 → → → → → if sL==L: CR LF
79 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CR LF
80 → → → → → instruct.append(0) CR LF
81 → → → → → elif sL<L: CR LF
82 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CR LF
83 → → → → → instruct.append(-1) CR LF
84 → → → → → elif sL>L: CR LF
85 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CR LF
86 → → → → → instruct.append(1) CR LF
87 → → → → → elif (c>120): CR LF
88 → → → → → if (d-b)>0 and (c-a)>0: CR LF
89 → → → → → L=(200-c)*(d-b)/(c-a) CR LF
90 → → → → → sL=abs(400-d-L) CR LF
91 → → → → → if sL==L: CR LF
92 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CR LF
93 → → → → → instruct.append(0) CR LF
94 → → → → → elif sL<L: CR LF
95 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CR LF
96 → → → → → instruct.append(1) CR LF
97 → → → → → elif sL>L: CR LF
98 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CR LF
99 → → → → → instruct.append(-1) CR LF
100 → → → → → else: CR LF
101 → → → → → if (scene_info.platform[0]+20)<(scene_info.ball[0]+2.5): CR LF
102 → → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CR LF
103 → → → → → instruct.append(1) CR LF
104 → → → → → elif (scene_info.platform[0]+20)>(scene_info.ball[0]+2.5): CR LF
→ → → → → comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CR LF

```

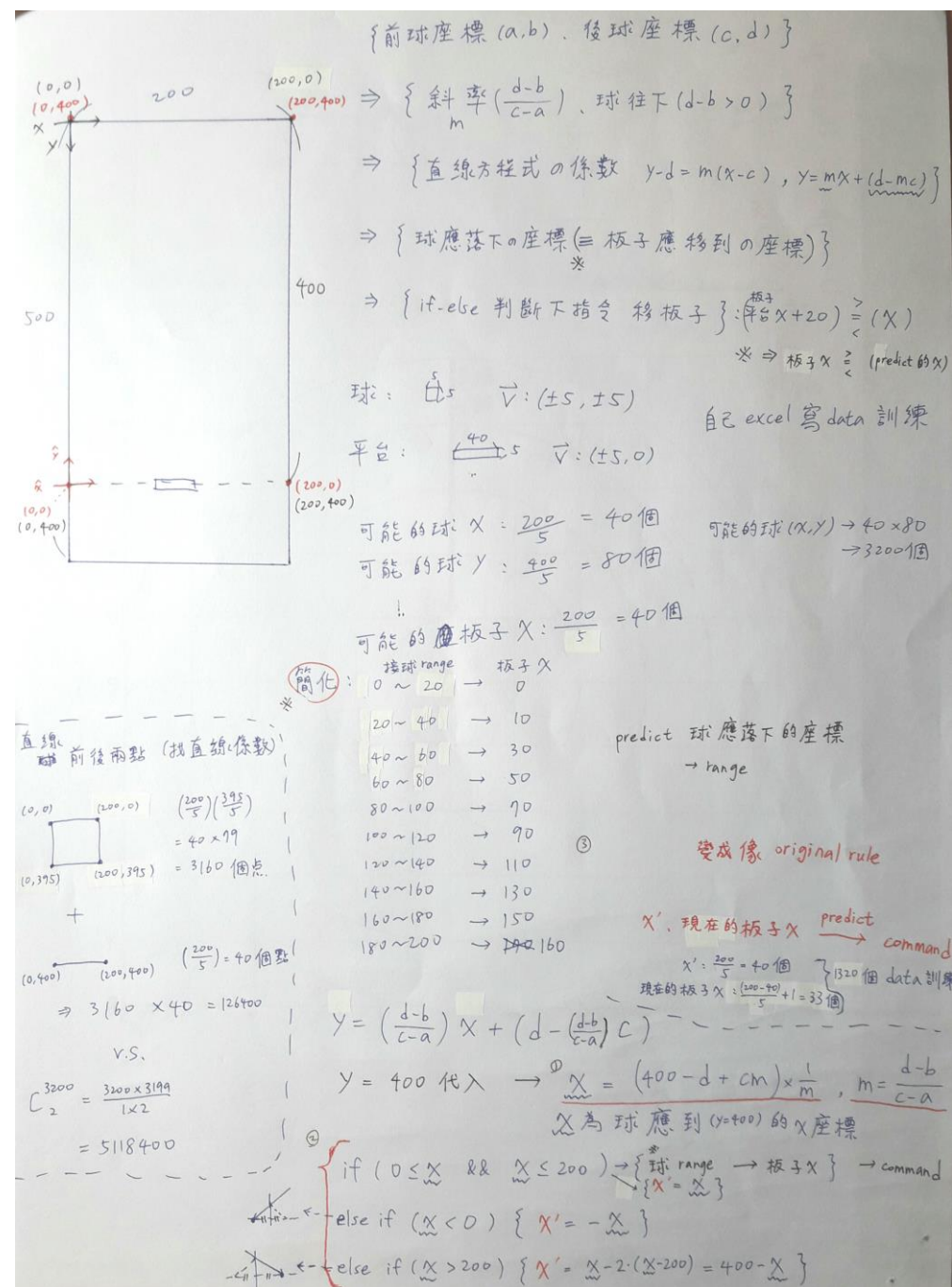
```

96         elif sL>L: CRLF
97         comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CRLF
98         instruct.append(-1) CRLF
99     else: CRLF
100     if (scene_info.platform[0]+20)<(scene_info.ball[0]+2.5): CRLF
101     comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CRLF
102     instruct.append(1) CRLF
103     elif (scene_info.platform[0]+20)>(scene_info.ball[0]+2.5): CRLF
104     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CRLF
105     instruct.append(-1) CRLF
106     else: CRLF
107     comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CRLF
108     instruct.append(0) CRLF
109     elif (d-b)>0 and (c-a)>0: CRLF
110     LL=2*(200-c)*(d-b)/(c-a) CRLF
111     if LL<400-d+3 and LL>400-d-3: CRLF
112     comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CRLF
113     instruct.append(0) CRLF
114     elif LL<400-d: CRLF
115     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CRLF
116     instruct.append(-1) CRLF
117     elif LL>400-d: CRLF
118     comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CRLF
119     instruct.append(1) CRLF
120     elif (d-b)>0 and (c-a)<0: CRLF
121     LL=2*c*(d-b)/(a-c) CRLF
122     if LL<400-d+3 and LL>400-d-3: CRLF
123     comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CRLF
124     instruct.append(0) CRLF
125     elif LL>400-d: CRLF
126     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CRLF
127     instruct.append(-1) CRLF
128     elif LL<400-d: CRLF
129     comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CRLF
130     instruct.append(1) CRLF
131

```

# 基本想法、做法 -- rule

- 打磚塊過渡打乒乓時，依公式(兩點和斜率構成直線方程式)計算球的落點，來做移動。





```

73  -> -> -> # 3.3. Put the code here to handle the scene information
74  -> -> -> a=c
75  -> -> -> b=d
76  -> -> -> c=scene_info.ball[0]
77  -> -> -> d=scene_info.ball[1]
78  -> -> -> if c-a==0:
79  -> -> ->     m=((d-b)/1)
80  -> -> -> else:
81  -> -> ->     m=((d-b)/(c-a))
82  -> -> -> # m=((d-b)/(c-a)) ...--> exception: divided by zero
83  -> -> -> L=(395-d+c*m)/m # L=(400-d+c*m)/m
84  -> -> -> LL.append(L)
85  -> -> -> Bx.append(c)
86  -> -> -> By.append(d)
87  -> -> -> Bx_subtract.append((c-a))
88  -> -> -> By_subtract.append((d-b))
89  -> -> -> px=0
90  -> -> -> ### inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], sce
91  -> -> -> ### inp_temp=np.array([L, scene_info.platform[0]])
92  -> -> ->
93  -> -> -> # inp_temp=np.array([c,d,(c-a),(d-b)])
94  -> -> -> # input=input_temp[np.newaxis,:])
95  -> -> ->
96  -> -> -> # 3.4. Send the instruction for this frame to the game process...
97  -> -> -> if L>=0 and L<=200:
98  -> -> ->     L=L
99  -> -> -> elif L<0:
100 -> -> ->     L=-L
101 -> -> ->     num=L//200
102 -> -> ->     rr=L%200
103 -> -> -> if num%2==1:
104 -> -> ->     L=200-rr
105 -> -> -> else:
106 -> -> ->     L=rr
107 -> -> -> elif L>200:
108 -> -> ->     L=L-200
109 -> -> ->     num=L//200

```

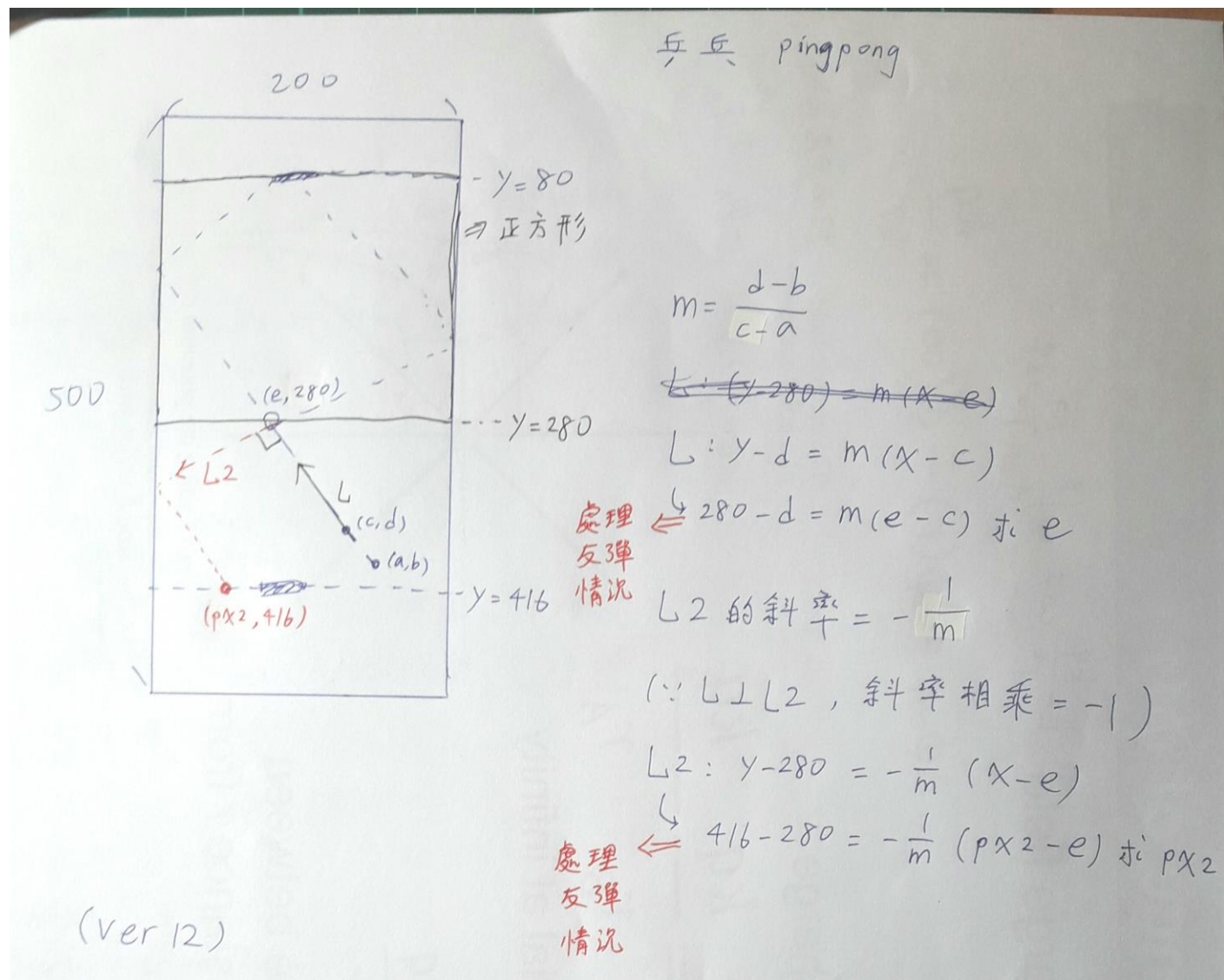
```
107 elif L>200:
108     L = L-200
109     num = L//200
110     rr = L%200
111     if num%2==1:
112         L = rr
113     else:
114         L = 200-rr
115
116 LL2.append(L)
117
118 # if L>=0 and L<=200:
119 #     L = L
120 # elif L<0 and L>-200:
121 #     L = -L
122 # elif L>200 and L<400:
123 #     L = 400-L
124
125 if L>=0 and L<20:
126     px=0
127 elif L>=20 and L<40:
128     px=10
129 elif L>=40 and L<60:
130     px=30
131 elif L>=60 and L<80:
132     px=50
133 elif L>=80 and L<100:
134     px=70
135 elif L>=100 and L<120:
136     px=90
137 elif L>=120 and L<140:
138     px=110
139 elif L>=140 and L<160:
140     px=130
141 elif L>=160 and L<180:
142     px=150
143 elif L>=180 and L<200:
```

```
143 elif·L>=180·and·L<200:CRLF
144     px=160CRLF
145     CRLF
146     CRLF
147     if·scene_info.platform[0]<px:CRLF
148         comm.send_instruction(scene_info.frame,·GameInstruction.CMD_RIGHT)CRLF
149         instruct.append(1)CRLF
150     elif·scene_info.platform[0]>px:CRLF
151         comm.send_instruction(scene_info.frame,·GameInstruction.CMD_LEFT)CRLF
152         instruct.append(-1)CRLF
153     else:CRLF
154         comm.send_instruction(scene_info.frame,·GameInstruction.CMD_NONE)CRLF
155         instruct.append(0)CRLF
156     CRLF
157     #·if·load_model.predict(input)==1:CRLF
158         #·comm.send_instruction(scene_info.frame,·GameInstruction.CMD_RIGHT)CRLF
159         #·instruct.append(1)CRLF
160     #·elif·load_model.predict(input)==-1:CRLF
161         #·comm.send_instruction(scene_info.frame,·GameInstruction.CMD_LEFT)CRLF
162         #·instruct.append(-1)CRLF
163     #·else:CRLF
164         #·comm.send_instruction(scene_info.frame,·GameInstruction.CMD_NONE)CRLF
165         #·instruct.append(0)CRLF
```



# 基本想法、做法 -- rule

- 打乒乓中後期，依多次公式和數學系同學分享的正方形內反彈原理，球上升下降都可計算球的落點，來做移動。



```
170  -> -> -> # 3.3 Put the code here to handle the scene information LF
171  -> -> -> a=c LF
172  -> -> -> b=d LF
173  -> -> -> c=scene_info.ball[0] LF
174  -> -> -> d=scene_info.ball[1] LF
175  -> -> -> if c-a==0: LF
176  -> -> ->     m=((d-b)/1) LF
177  -> -> -> else: LF
178  -> -> ->     m=((d-b)/(c-a)) LF
179  -> -> -> LF
180  -> -> -> # 3.4 Send the instruction for this frame to the game process LF
181  -> -> -> if d-b>0: LF
182  -> -> ->     L=(415-d+c*m)/m LF
183  -> -> ->     LF
184  -> -> ->     if L>=0 and L<=200: LF
185  -> -> ->         L=.L LF
186  -> -> ->     elif L<0: LF
187  -> -> ->         L=-.L LF
188  -> -> ->         num=.L//200 LF
189  -> -> ->         rr=.L%200 LF
190  -> -> ->         if num%2==1: LF
191  -> -> ->             L=.200-rr LF
192  -> -> ->         else: LF
193  -> -> ->             L=.rr LF
194  -> -> ->     elif L>200: LF
195  -> -> ->         L=.L-200 LF
196  -> -> ->         num=.L//200 LF
197  -> -> ->         rr=.L%200 LF
198  -> -> ->         if num%2==1: LF
199  -> -> ->             L=.rr LF
200  -> -> ->         else: LF
201  -> -> ->             L=.200-rr LF
202  -> -> ->     LF
203  -> -> ->     if scene_info.platform_2P[0]+20>L+3:..#+4 LF
204  -> -> ->         comm.send_instruction(scene_info.frame, PlatformAction.MOVE_LEFT) LF
205  -> -> ->     elif scene_info.platform_2P[0]+20<L+2:..#+1 LF
206  -> -> ->         comm.send_instruction(scene_info.frame, PlatformAction.MOVE_RIGHT) LF
```

```
207 else: LF
208     comm.send_instruction(scene_info.frame, PlatformAction.NONE) LF
209 else: LF
210 if d>280: LF
211     L=(280-d+c*m)/m LF
212     LF
213 if L>=0 and L<=200: LF
214     L=L LF
215 elif L<0: LF
216     L=-L LF
217     num=L//200 LF
218     rr=L%200 LF
219     if num%2==1: LF
220         L=200-rr LF
221     else: LF
222         L=rr LF
223 elif L>200: LF
224     L=L-200 LF
225     num=L//200 LF
226     rr=L%200 LF
227     if num%2==1: LF
228         L=rr LF
229     else: LF
230         L=200-rr LF
231     LF
232     m2=(-1/m) LF
233     #L2=(416-280+L*m2)/m2 LF
234     L2=(415-280+L*m2)/m2 LF
235     LF
236 if L2>=0 and L2<=200: LF
237     L2=L2 LF
238 elif L2<0: LF
239     L2=-L2 LF
240     num=L2//200 LF
241     rr=L2%200 LF
242     if num%2==1: LF
243         L2=200-rr LF
```

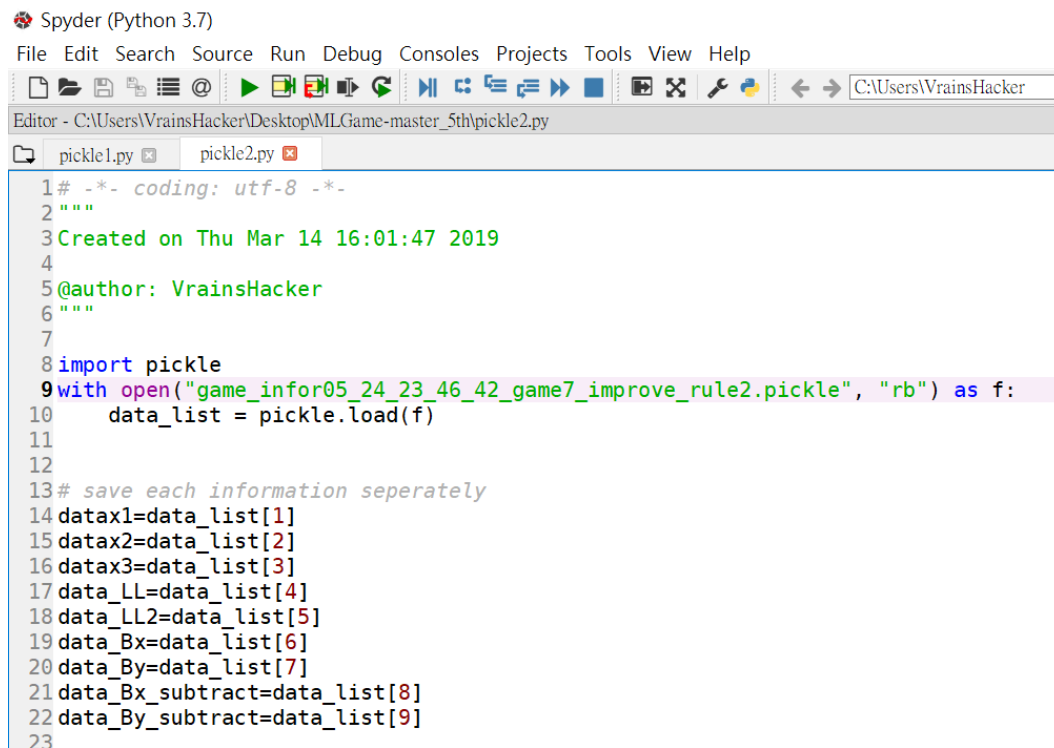
```

238 elif L2<0:LF
239     L2.=-L2LF
240     num.=L2//200LF
241     rr.=L2%200LF
242     if num%2==1:LF
243         L2.=200-rrLF
244     else:LF
245         L2.=rrLF
246     elif L2>200:LF
247         L2.=L2-200LF
248         num.=L2//200LF
249         rr.=L2%200LF
250         if num%2==1:LF
251             L2.=rrLF
252         else:LF
253             L2.=200-rrLF
254     LF
255     LL2=L2LF
256     LF
257     if scene_info.platform_2P[0]+20>L2+3:..#+4..+2.5LF
258         comm.send_instruction(scene_info.frame,PlatformAction.MOVE_LEFT)LF
259     elif scene_info.platform_2P[0]+20<L2+2:..#+1..+2.5LF
260         comm.send_instruction(scene_info.frame,PlatformAction.MOVE_RIGHT)LF
261     else:LF
262         comm.send_instruction(scene_info.frame,PlatformAction.NONE)LF
263     else:LF
264     if scene_info.platform_2P[0]+20>LL2+3:..#+4..+2.5LF
265         comm.send_instruction(scene_info.frame,PlatformAction.MOVE_LEFT)LF
266     elif scene_info.platform_2P[0]+20<LL2+2:..#+1..+2.5LF
267         comm.send_instruction(scene_info.frame,PlatformAction.MOVE_RIGHT)LF
268     else:LF
269         comm.send_instruction(scene_info.frame,PlatformAction.NONE)LF

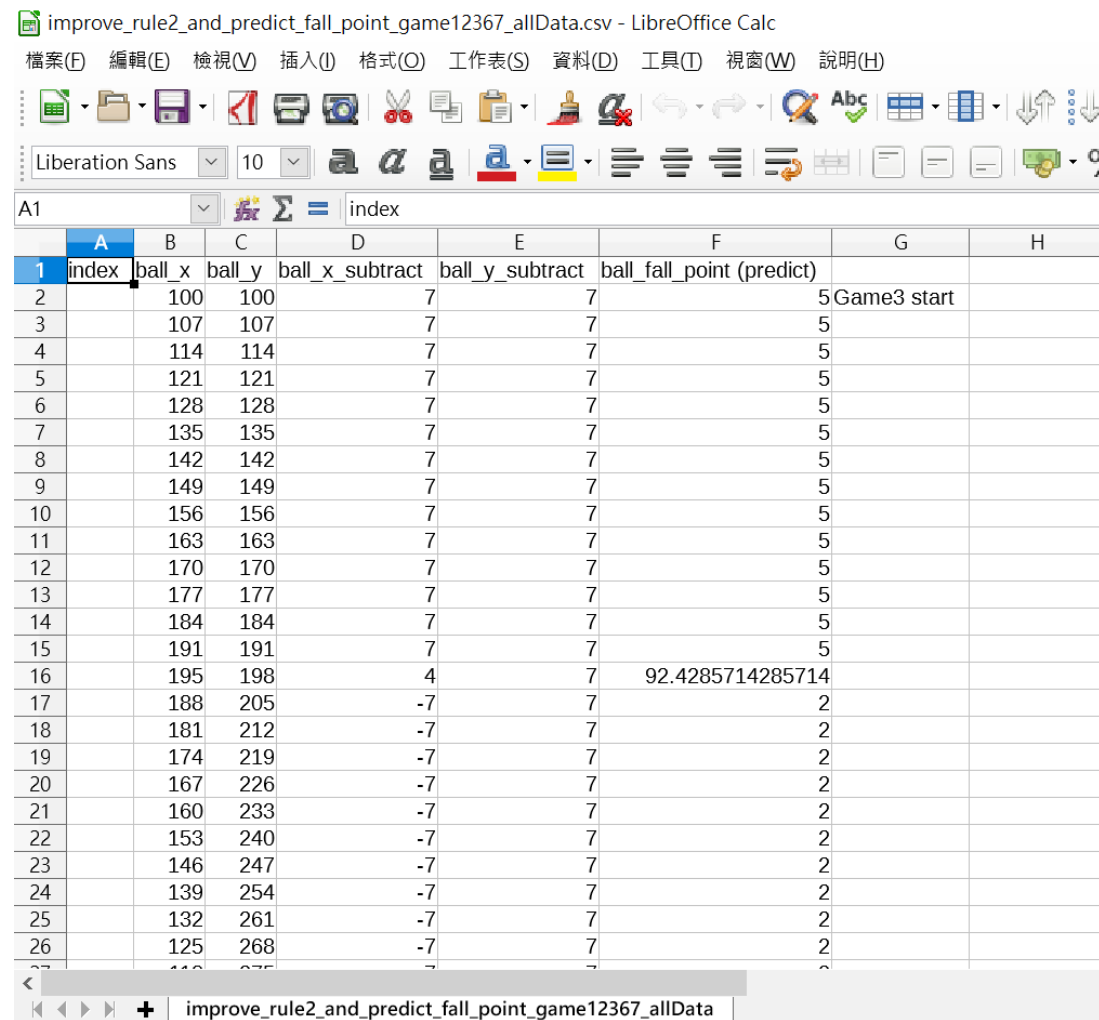
```

# 基本想法、做法 -- 額外技巧?

- 用pickle取data放進.csv檔去train
- For example:



```
1# -*- coding: utf-8 -*-
2"""
3Created on Thu Mar 14 16:01:47 2019
4
5@author: VbrainsHacker
6"""
7
8import pickle
9with open("game_infor05_24_23_46_42_game7_improve_rule2.pickle", "rb") as f:
10    data_list = pickle.load(f)
11
12
13# save each information seperately
14datax1=data_list[1]
15datax2=data_list[2]
16datax3=data_list[3]
17data_LL=data_list[4]
18data_LL2=data_list[5]
19data_Bx=data_list[6]
20data_By=data_list[7]
21data_Bx_subtract=data_list[8]
22data_By_subtract=data_list[9]
23
```



improve\_rule2\_and\_predict\_fall\_point\_game12367\_allData.csv - LibreOffice Calc

檔案(F) 編輯(E) 檢視(V) 插入(I) 格式(O) 工作表(S) 資料(D) 工具(T) 視窗(W) 說明(H)

Liberation Sans 10

A1	A	B	C	D	E	F	G	H
1	index	ball_x	ball_y	ball_x_subtract	ball_y_subtract	ball_fall_point (predict)		
2		100	100	7	7	5	Game3 start	
3		107	107	7	7	5		
4		114	114	7	7	5		
5		121	121	7	7	5		
6		128	128	7	7	5		
7		135	135	7	7	5		
8		142	142	7	7	5		
9		149	149	7	7	5		
10		156	156	7	7	5		
11		163	163	7	7	5		
12		170	170	7	7	5		
13		177	177	7	7	5		
14		184	184	7	7	5		
15		191	191	7	7	5		
16		195	198	4	7	92.4285714285714		
17		188	205	-7	7	2		
18		181	212	-7	7	2		
19		174	219	-7	7	2		
20		167	226	-7	7	2		
21		160	233	-7	7	2		
22		153	240	-7	7	2		
23		146	247	-7	7	2		
24		139	254	-7	7	2		
25		132	261	-7	7	2		
26		125	268	-7	7	2		

improve\_rule2\_and\_predict\_fall\_point\_game12367\_allData



# 基本想法、做法 -- model & features

- 一開始打磚塊用knn，  
feature為球的x座標和平  
台的x座標，預測移動指  
令。

	A	B	C	D
1	index	X(200)	plate_x	instruction
2			0	75
3			0	70
4			0	65
5			0	60
6			0	55
7			0	50
8			0	45
9			0	40
10			0	35
11			0	30
12			0	25
13			0	20
14			0	15
15			0	10
16		89.5714285714286	5	1
17			7	10
18			7	5
19			7	0
20			7	0
21			7	0
22			7	0
23			7	0
24			7	0
25			7	0

```
new_game12367_load_and_train_knn.py
6  """
7  """
8  import pandas as pd #for handling .csv files
9  import numpy as np
10 """
11 customer_data=pd.read_csv("game_12367_all_data.csv")
12 customer_data.head(10) #show first ten samples of data
13 """
14 x=customer_data.iloc[:,1:3].values
15 y=customer_data.iloc[:,3:4].values
16 """
17 """
18 from sklearn.model_selection import train_test_split
19 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
20 """
21 """train your model here
22 #from xxx import ooo_model
23 from sklearn.neighbors import KNeighborsClassifier
24 #ooo=ooo_model()
25 neigh = KNeighborsClassifier(n_neighbors=1)
26 #ooo.fit(x_train,y_train)
27 #neigh.fit(x_train,y_train)
28 neigh.fit(x,y)
29 #ooo.predict(x_test)
30 y_knn=neigh.predict(x)
31 #check the acc to see how well you've trained the model
32 #acc=?
33 from sklearn.metrics import accuracy_score
34 acc=accuracy_score(y_knn,y)
35 """
36 """
37 """save model
38 import pickle
39 """
40 #filename="ooo_example0401.sav"
41 filename="neigh_knn_game12367_0523.sav"
42 pickle.dump(neigh, open(filename, 'wb'))
```

Python file length : 1.222 lines : 48

# 基本想法、做法 -- model & features

- 打磚塊過渡打乒乓時用 random forest，feature 為球的x座標、球的y座標、現在球的x座標減掉前一個球的x座標、現在球的y座標減掉前一個球的y座標，預測球的落點。

	A	B	C	D	E	F
1	index	ball_x	ball_y	ball_x_subtract	ball_y_subtract	ball_fall_point (predict)
2		100	100	7	7	5
3		107	107	7	7	5
4		114	114	7	7	5
5		121	121	7	7	5
6		128	128	7	7	5
7		135	135	7	7	5
8		142	142	7	7	5
9		149	149	7	7	5

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - C:\Users\WraingsHacker\Desktop\5月24研究 improve rule2 and predict 落點\improve\_rule2\_game12367\_load\_and\_train\_random\_forest\_tree.py

```
pickle1.py x pickle2.py x improve_rule2_game12367_load_and_train_random_forest_tree.py* x
7
8 import pandas as pd #for handling .csv files
9 import numpy as np
10
11 customer_data = pd.read_csv("improve_rule2_and_predict_fall_point_game12367_allData.csv")
12 customer_data.head(10) # show first ten samples of data
13
14 x=customer_data.iloc[:,1:5].values
15 y=customer_data.iloc[:,5:6].values
16
17
18 from sklearn.model_selection import train_test_split
19 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
20
21 from sklearn.ensemble import RandomForestRegressor
22 from sklearn.metrics import mean_squared_error
23
24 forest_reg = RandomForestRegressor(random_state=0)
25 forest_reg.fit(x,y)
26 y_predict = forest_reg.predict(x)
27
28 MSE=mean_squared_error(y,y_predict)
29 RMSE=np.sqrt(MSE)
30 print("MSE: %f " % MSE)
31 print("RMSE: %f " % RMSE)
32
33
34 ### save model
35 import pickle
36
37 #filename="ooo_example0401.sav"
38 filename="forest_reg_random_forest_tree_game12367_improve_0524.sav"
39 pickle.dump(forest_reg, open(filename, 'wb'))
40
41 ### load model
```

# 基本想法、做法 -- model & features

- 打乒乓中後期用SVM，  
feature為球的x座標、球的y座標、現在球的x座標減掉前一個球的x座標、現在球的y座標減掉前一個球的y座標，預測球的落點。

	B	C	D	E	F
1	C	D	CsubtractA	DsubtractB	Lx
2	75	100	-25	-150	71.66666666666667
3	82	107	7	7	75
4	89	114	7	7	75
5	96	121	7	7	75
6	103	128	7	7	75
7	110	135	7	7	75
8	117	142	7	7	75
9	124	149	7	7	75

```
pingpong_load_and_train_pl_svm_ver3.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sun Apr 14 18:53:41 2019
4
5 @author: VrainHacker
6 """
7
8 import pandas as pd #for handling .csv files
9 import numpy as np
10
11 customer_data = pd.read_csv("pingpong_data_pl.csv")
12 customer_data.head(10) # show first ten samples of data
13
14 x = customer_data.iloc[1:, 1:5].values
15 y = customer_data.iloc[1:, 5:6].values
16
17
18
19 from sklearn.svm import SVR
20 regressor = SVR(gamma=0.005, C=20, epsilon=0.3)
21 # gamma=0.01, C=25, epsilon=0.2 --> poor
22 # gamma=0.00075, C=15, epsilon=0.4
23 # gamma=0.001, C=20, epsilon=0.3 ... # gamma=0.0005, C=10, epsilon=0.5
24 regressor.fit(x, y)
25
26 y_predict = regressor.predict(x)
27 from sklearn.metrics import mean_squared_error
28 MSE = mean_squared_error(y, y_predict)
29 RMSE = np.sqrt(MSE)
30 print("MSE: %f" % MSE)
31 print("RMSE: %f" % RMSE)
```

# SVM

- svm ver20       $\gamma=0.005, C=1000, \epsilon=0.2$
- svm ver22       $\gamma=0.001, C=20, \epsilon=0.3$
- svm ver23       $\gamma=0.005, C=20, \epsilon=0.3$

# 勝負表

勝負：

rule ver 15:

① ver 15 輸給 ver 20

② ver 15 平手 ver 22 → P1 時贏, P2 時輸

→ 0 勝 1 輸 3 平

③ ver 15 平手 ver 19 → P2 時贏, P1 時輸

④ ver 15 平手 ver 23 → P1 時贏, P2 時輸

random forest ver 19:

① ver 19 輸給 ver 20

② ver 19 平手 ver 15 → P2 時贏, P1 時輸

→ 1 勝 1 輸 2 平

③ ver 19 平手 ver 23 → P2 時贏, P1 時輸 → speed 30

④ ver 19 打敗 ver 22

svm-1 ver 20:

① ver 20 打敗 ver 19

② ver 20 平手 ver 22 → P2 時贏, P1 時輸

→ 2 勝 0 輸 2 平

③ ver 20 打敗 ver 15

④ ver 20 平手 ver 23 → P1 時贏, P2 時輸

svm-2 ver 22:

① ver 22 平手 ver 20 → P2 時贏, P1 時輸

② ver 22 平手 ver 15 → P1 時贏, P2 時輸

→ 0 勝 1 輸 3 平

③ ver 22 平手 ver 23 → P2 時贏, P1 時輸

④ ver 22 輸給 ver 19 → speed 28

svm-3 ver 23:

① ver 23 平手 ver 22 → P2 時贏, P1 時輸 → speed 28

② ver 23 平手 ver 20 → P1 時贏, P2 時輸

③ ver 23 平手 ver 19 → P2 時贏, P1 時輸 → speed 30

→ 4 平

④ ver 23 平手 ver 15 → P1 時贏, P2 時輸

- 考慮許久，最後決定用四次平手的 svm ver23，感覺較能彈性應對狀況。



# 心得感想

- 實作為主，理論為輔。學習過程很有趣，不像痛苦的考試。
- 每個階段的目標明確，自己彈性安排時間，不必擠在期末。
- (未來視)覺得自己做得出來，便樂意投入時間。卡關時可先放下，隨時有空時便可思考解法。
- 有教授費心準備的簡報可看，遇到問題時能問助教，同學的分享更是讓我大開眼界，同學們的想法是我所沒想到的。因此，就算我沒天分又單獨一人為一組，最後也能努力地順利完成課堂的最低要求。(跟物聯網小車車那時很不一樣～)
- 謝謝教授、助教、同學。