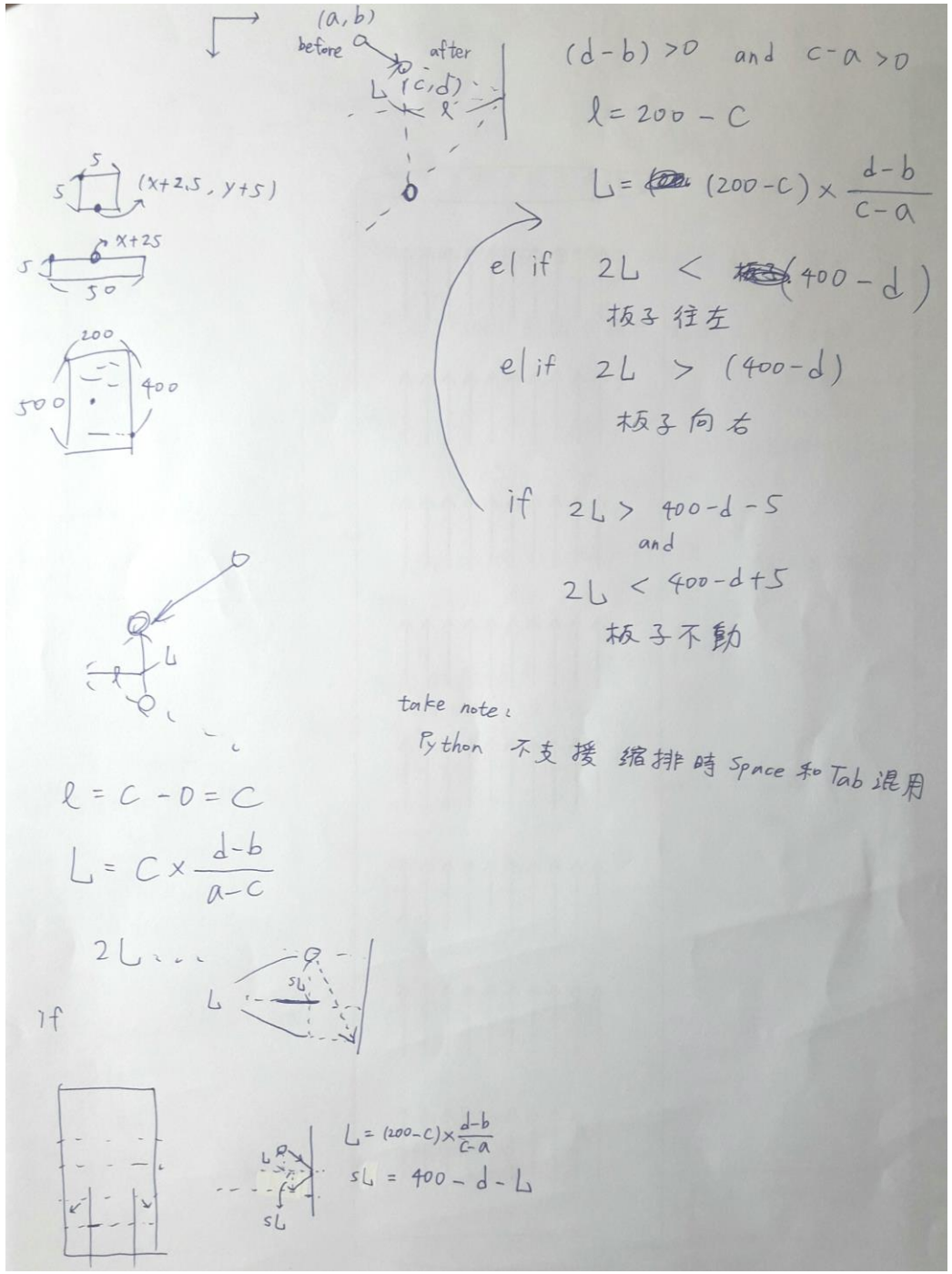
基於遊戲的機器學習入門 Final 報告

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。



ml_play_original_rule_pass_pass_game1.py

```python
        # 3.4. Send the instruction for this frame to the game process
        #if scene_info.frame>1300 and scene_info.frame<1500:
        #    comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
        #    instruct.append(-1)
        if (400-d)<100:
            if (c<80):
                if (d-b)>0 and (c-a)<0 :
                    L=c*(d-b)//(a-c)
                    sL=abs(400-d-L)
                    if sL==L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
                        instruct.append(0)
                    elif sL<L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                        instruct.append(-1)
                    elif sL>L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                        instruct.append(1)
            elif (c>120):
                if (d-b)>0 and (c-a)>0 :
                    L=(200-c)*(d-b)//(c-a)
                    sL=abs(400-d-L)
                    if sL==L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
                        instruct.append(0)
                    elif sL<L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                        instruct.append(1)
                    elif sL>L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                        instruct.append(-1)
            else:
                if (scene_info.platform[0]+20)<(scene_info.ball[0]+2.5):
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                    instruct.append(1)
                elif (scene_info.platform[0]+20)>(scene_info.ball[0]+2.5):
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
```

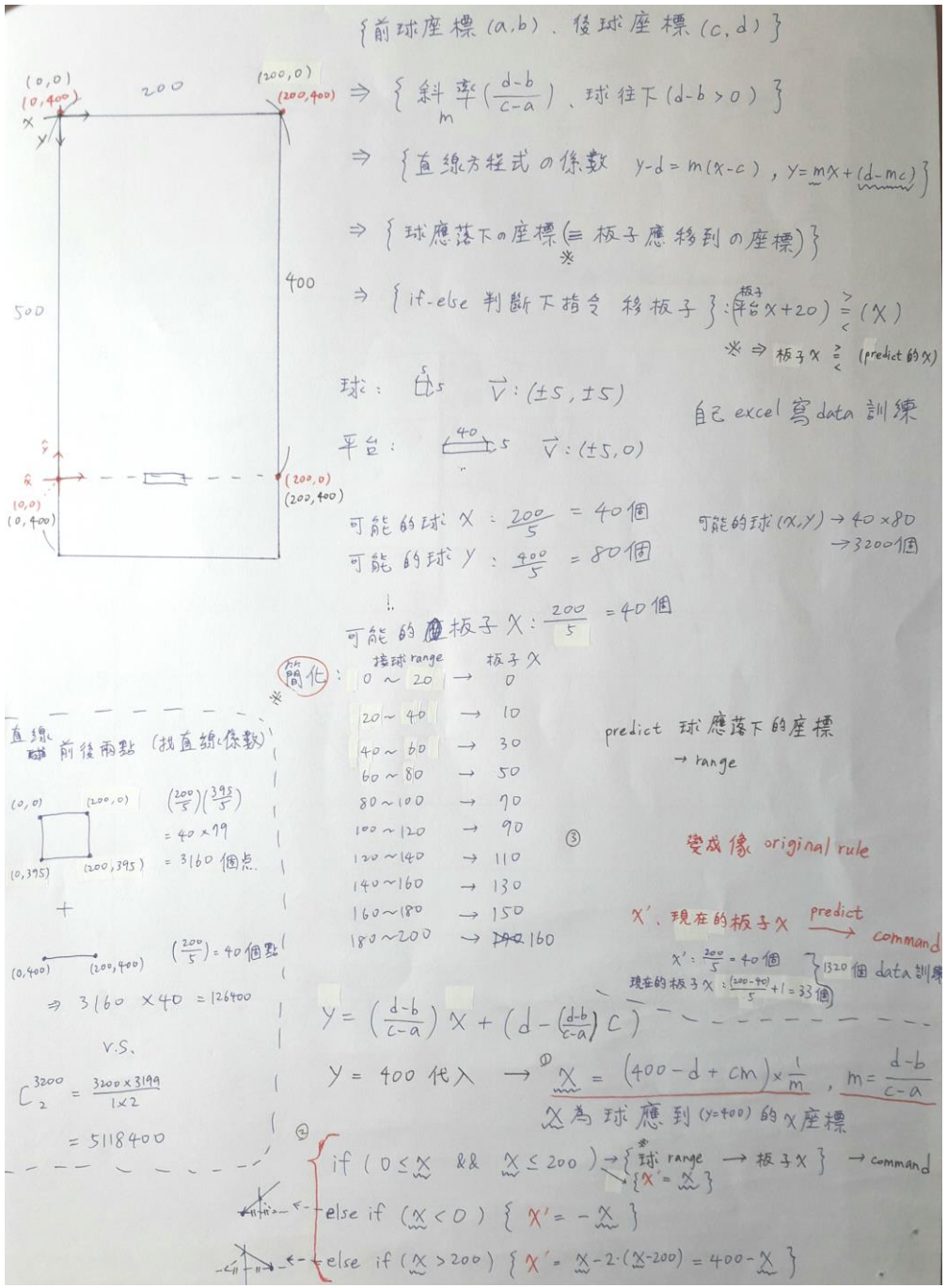
ml_play_original_rule_pass_pass_game1.py

```python
                    elif sL>L:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                        instruct.append(-1)
                else:
                    if (scene_info.platform[0]+20)<(scene_info.ball[0]+2.5):
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                        instruct.append(1)
                    elif (scene_info.platform[0]+20)>(scene_info.ball[0]+2.5):
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                        instruct.append(-1)
                    else:
                        comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
                        instruct.append(0)
            elif (d-b)>0 and (c-a)>0 :
                LL=2*(200-c)*(d-b)//(c-a)
                if LL<400-d+3 and LL>400-d-3 :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
                    instruct.append(0)
                elif LL<400-d :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                    instruct.append(-1)
                elif LL>400-d :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                    instruct.append(1)
            elif (d-b)>0 and (c-a)<0 :
                LL=2*c*(d-b)//(a-c)
                if LL<400-d+3 and LL>400-d-3 :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
                    instruct.append(0)
                elif LL>400-d :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
                    instruct.append(-1)
                elif LL<400-d :
                    comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
                    instruct.append(1)
```

Python file | length : 4,738 lines : 131 | Ln : 1 Col :

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。



ml_play_L_y=395_and_improve_rule2_by_c-a!=0.py

```python
        # 3.3. Put the code here to handle the scene information
        a=c
        b=d
        c=scene_info.ball[0]
        d=scene_info.ball[1]
        if c-a==0:
            m=((d-b)/1)
        else:
            m=((d-b)/(c-a))
        # m=((d-b)/(c-a))    --> exception: divided by zero
        L=(395-d+c*m)/m  #L=(400-d+c*m)/m
        LL.append(L)
        Bx.append(c)
        By.append(d)
        Bx_subtract.append((c-a))
        By_subtract.append((d-b))
        px=0
        ### inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], sce
        ###inp_temp=np.array([L, scene_info.platform[0]])
        
        #inp_temp=np.array([c,d,(c-a), (d-b)])
        #input=inp_temp[np.newaxis, :]

        # 3.4. Send the instruction for this frame to the game process
        if L>=0 and L<=200:
            L = L
        elif L<0:
            L = -L
            num = L//200
            rr = L%200
            if num%2==1:
                L = 200-rr
            else:
                L = rr
        elif L>200:
            L = L-200
            num = L//200
```

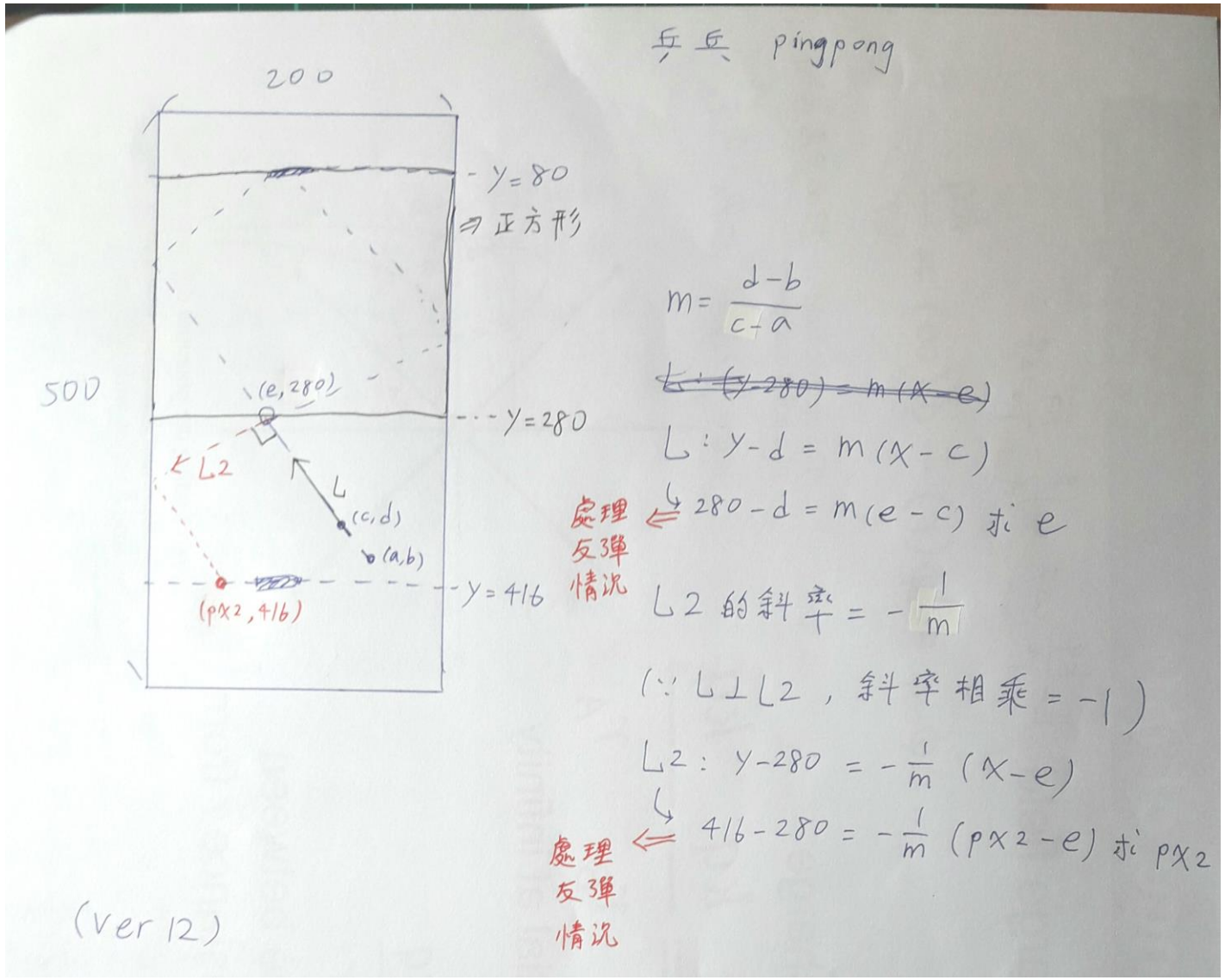
ml_play_L_y=395_and_improve_rule2_by_c-a!=0.py

```
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:
```

```
        elif L>=180 and L<200:
            px=160

        if scene_info.platform[0]<px:
            comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
            instruct.append(1)
        elif scene_info.platform[0]>px:
            comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
            instruct.append(-1)
        else:
            comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
            instruct.append(0)

        # if load_model.predict(input)==1:
            # comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)
            # instruct.append(1)
        # elif load_model.predict(input)==-1:
            # comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)
            # instruct.append(-1)
        # else:
            # comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)
            # instruct.append(0)
```

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。



ml_play_ver15_simplify_from_ver12.py

```
        # 3.3 Put the code here to handle the scene information
            a=c
            b=d
            c=scene_info.ball[0]
            d=scene_info.ball[1]
            if c-a==0:
                m=((d-b)/1)
            else:
                m=((d-b)/(c-a))

        # 3.4 Send the instruction for this frame to the game process
            if d-b>0:
                L=(415-d+c*m)/m

                if L>=0 and L<=200:
                    L = L
                elif L<0:
                    L = -L
                    num = L//200
                    rr = L%200
                    if num%2==1:
                        L = 200-rr
                    else:
                        L = rr
                elif L>200:
                    L = L-200
                    num = L//200
                    rr = L%200
                    if num%2==1:
                        L = rr
                    else:
                        L = 200-rr

                if scene_info.platform_2P[0]+20>L+3:  #+4
                    comm.send_instruction(scene_info.frame, PlatformAction.MOVE_LEFT)
                elif scene_info.platform_2P[0]+20<L+2:  #+1
                    comm.send_instruction(scene_info.frame, PlatformAction.MOVE_RIGHT)
```

ml_play_ver15_simplify_from_ver12.py

```
            else:
                comm.send_instruction(scene_info.frame, PlatformAction.NONE)
        else:
            if d>280:
                L=(280-d+c*m)/m

                if L>=0 and L<=200:
                    L = L
                elif L<0:
                    L = -L
                    num = L//200
                    rr = L%200
                    if num%2==1:
                        L = 200-rr
                    else:
                        L = rr
                elif L>200:
                    L = L-200
                    num = L//200
                    rr = L%200
                    if num%2==1:
                        L = rr
                    else:
                        L = 200-rr

                m2=(-1/m)
                #L2=(416-280+L*m2)/m2
                L2=(415-280+L*m2)/m2

                if L2>=0 and L2<=200:
                    L2 = L2
                elif L2<0:
                    L2 = -L2
                    num = L2//200
                    rr = L2%200
                    if num%2==1:
                        L2 = 200-rr
```

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

基本想法、做法 -- 額外技巧?

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

Spyder (Python 3.7)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\VrainsHacker

Editor - C:\Users\VrainsHacker\Desktop\MLGame-master_5th\pickle2.py

pickle1.py pickle2.py

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 14 16:01:47 2019

@author: VrainsHacker
"""

import pickle
with open("game_infor05_24_23_46_42_game7_improve_rule2.pickle", "rb") as f:
    data_list = pickle.load(f)


# save each information seperately
datax1=data_list[1]
datax2=data_list[2]
datax3=data_list[3]
data_LL=data_list[4]
data_LL2=data_list[5]
data_Bx=data_list[6]
data_By=data_list[7]
data_Bx_subtract=data_list[8]
data_By_subtract=data_list[9]

```

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 index

	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，feaure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	-1
3		0	70	-1
4		0	65	-1
5		0	60	-1
6		0	55	-1
7		0	50	-1
8		0	45	-1
9		0	40	-1
10		0	35	-1
11		0	30	-1
12		0	25	-1
13		0	20	-1
14		0	15	-1
15		0	10	-1
16		89.5714285714286	5	1
17		7	10	-1
18		7	5	-1
19		7	0	0
20		7	0	0
21		7	0	0
22		7	0	0
23		7	0	0
24		7	0	0
25		7	0	0

new_game12367_load_and_train_knn.py

```python
"""

import pandas as pd #for handling .csv files
import numpy as np

customer_data =pd.read_csv("game_12367_all_data.csv")
customer_data.head(10) # show first ten samples of data

x=customer_data.iloc[:,1:3].values
y=customer_data.iloc[:,3:4].values


from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

#%% train your model here
#from xxx import ooo_model
from sklearn.neighbors import KNeighborsClassifier
#ooo=ooo_model()
neigh = KNeighborsClassifier(n_neighbors=1)
#ooo.fit(x_train,y_train)
#neigh.fit(x_train,y_train)
neigh.fit(x,y)
#ooo.predict(x_test)
y_knn=neigh.predict(x)
# check the acc to see how well you've trained the model
#acc=?
from sklearn.metrics import accuracy_score
acc=accuracy_score(y_knn,y)


#%% save model
import pickle

#filename="ooo_example0401.sav"
filename="neigh_knn_game12367_0523.sav"
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，feaure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

C:\Users\VrainsHacker

Editor - C:\Users\VrainsHacker\Desktop\5月24研究 improve rule2 and predict 落點\improve_rule2_game12367_load_and_train_random_forest_tree.py

pickle1.py | pickle2.py | improve_rule2_game12367_load_and_train_random_forest_tree.py*

```python

import pandas as pd #for handling .csv files
import numpy as np

customer_data =pd.read_csv("improve_rule2_and_predict_fall_point_game12367_allData.csv")
customer_data.head(10) # show first ten samples of data

x=customer_data.iloc[:,1:5].values
y=customer_data.iloc[:,5:6].values


from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error

forest_reg = RandomForestRegressor(random_state=0)
forest_reg.fit(x,y)
y_predict = forest_reg.predict(x)

MSE=mean_squared_error(y,y_predict)
RMSE=np.sqrt(MSE)
print("MSE: %f " % MSE)
print("RMSE: %f " % RMSE)


#%% save model
import pickle

#filename="ooo_example0401.sav"
filename="forest_reg_random_forest_tree_game12367_improve_0524.sav"
pickle.dump(forest_reg, open(filename, 'wb'))

## load model
```

Permissions: RW End-

基本想法、做法 -- model & features

- 打乒乓中後期用SVM，feaure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	CsubstractA	DsubstractB	Lx
2	75	100	-25	-150	71.6666666666667
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

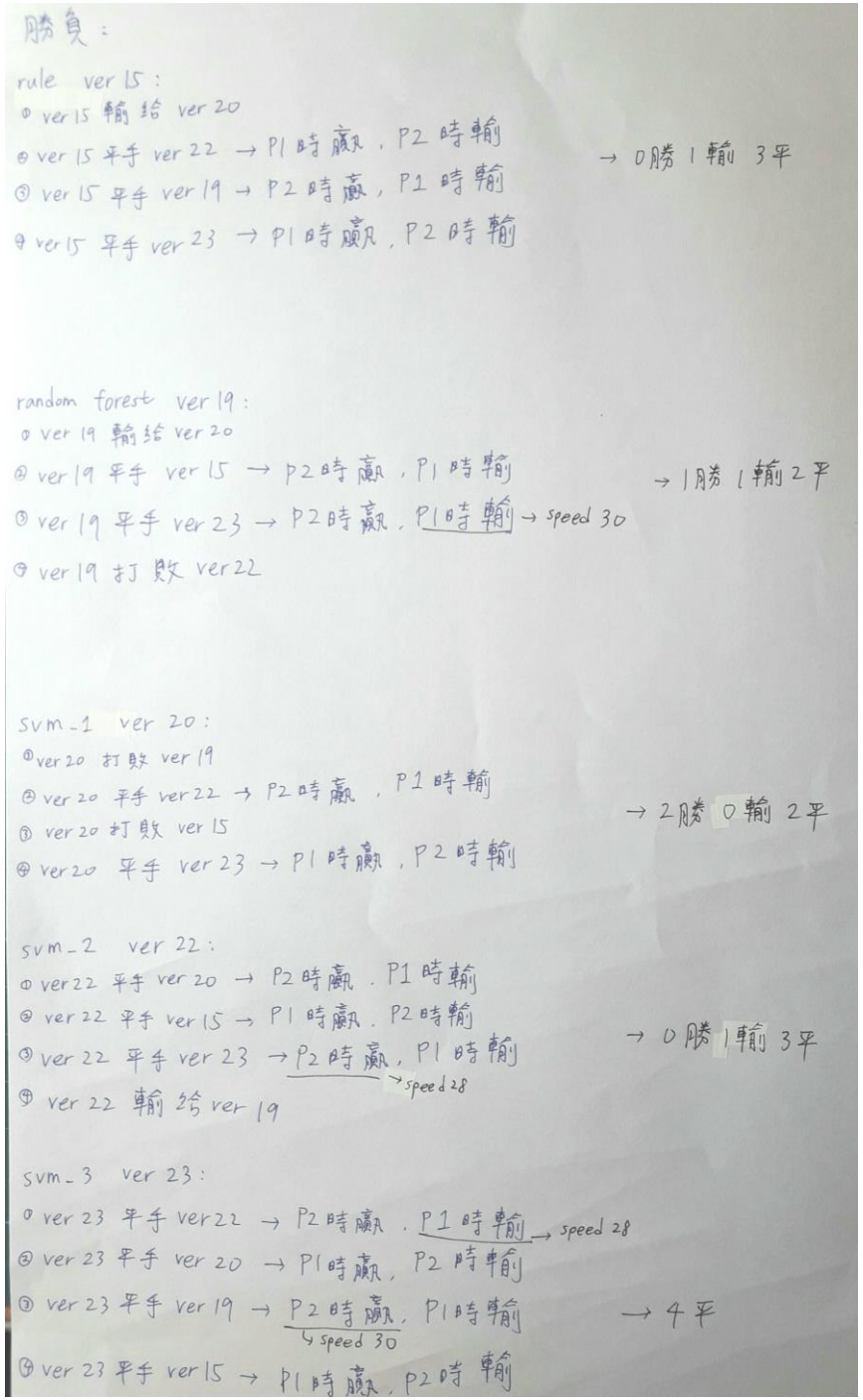
pingpong_load_and_train_p1_svm_ver3.py

```python
# -*- coding: utf-8 -*-
"""
Created on Sun Apr 14 18:53:41 2019

@author: VrainsHacker
"""

import pandas as pd #for handling .csv files
import numpy as np

customer_data =pd.read_csv("pingpong_data_p1.csv")
customer_data.head(10) # show first ten samples of data

x=customer_data.iloc[1:,1:5].values
y=customer_data.iloc[1:,5:6].values


from sklearn.svm import SVR
regressor=SVR(gamma=0.005,C=20,epsilon=0.3)  
#gamma=0.01,C=25,epsilon=0.2 -->poor
#gamma=0.00075,C=15,epsilon=0.4
#gamma=0.001,C=20,epsilon=0.3      #gamma=0.0005,C=10,epsilon=0.5
regressor.fit(x,y)

y_predict=regressor.predict(x)
from sklearn.metrics import mean_squared_error
MSE=mean_squared_error(y,y_predict)
RMSE=np.sqrt(MSE)
print("MSE: %f " % MSE)
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
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
→ 0勝 1輸 3平

random forest ver 19：
① ver 19 輸給 ver 20
② ver 19 平手 ver 15 → P2 時贏，P1 時輸
③ ver 19 平手 ver 23 → P2 時贏，P1 時輸 → speed 30
④ ver 19 打敗 ver 22
→ 1勝 1輸 2平

svm_1 ver 20：
① ver 20 打敗 ver 19
② ver 20 平手 ver 22 → P2 時贏，P1 時輸
③ ver 20 打敗 ver 15
④ ver 20 平手 ver 23 → P1 時贏，P2 時輸
→ 2勝 0輸 2平

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
③ ver 22 平手 ver 23 → P2 時贏，P1 時輸 → speed 28
④ ver 22 輸給 ver 19
→ 0勝 1輸 3平

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 ver 23 平手 ver 15 → P1 時贏，P2 時輸
→ 4平

- 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。