

基於遊戲的機器學習入門

4/11 報告

張君豪



Original Rule




打遊戲的規則

- 一開始球在左邊，平台就往左移；球在右邊，平台就往右移。發現平台跟不上球。
- 後來估算球反彈的程度在左右移動，減少不必要的移動，平台就接得到球。

資料的預處理

- ➡ 球的前後座標差
- ➡ 球離牆的水平距離
- ➡ 球離 $y=400$ 的垂直距離



觀察與發現

- ➡ 規則寫得好的話，能通關

code (` `)

```
C:\Users\VrainsHacker\Desktop\MLGame-master\arkanoid\ml\ml_play_original_rule_pass.py - Notepad++
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

ml_play.py x ml_play_original_rule_pass.py x
62  -> #.3.3..Put the code here to handle the scene informationCRLF
63  -> a=cCRLF
64  -> b=dCRLF
65  -> c=scene_info.ball[0]CRLF
66  -> d=scene_info.ball[1]CRLF
67  CRLF
68  -> #.3.4..Send the instruction for this frame to the game process.....CRLF
69  -> if scene_info.frame>1300:CRLF
70  ->     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)CRLF
71  ->     instruct.append(-1)CRLF
72  -> elif (400-d)<100:CRLF
73  ->     if (c<80):CRLF
74  ->         if (d-b)>0 and (c-a)<0:CRLF
75  ->             L=c*(d-b)/(a-c)CRLF
76  ->             sL=abs(400-d-L)CRLF
77  ->             if sL==L:CRLF
78  ->                 comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)CRLF
79  ->                 instruct.append(0)CRLF
80  ->             elif sL<L:CRLF
81  ->                 comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)CRLF
82  ->                 instruct.append(-1)CRLF
83  ->             elif sL>L:CRLF
84  ->                 comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)CRLF
85  ->                 instruct.append(1)CRLF
86  ->         elif (c>120):CRLF
87  ->             if (d-b)>0 and (c-a)>0:CRLF
88  ->                 L=(200-c)*(d-b)/(c-a)CRLF
89  ->                 sL=abs(400-d-L)CRLF
90  ->                 if sL==L:CRLF
91  ->                     comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE)CRLF
92  ->                     instruct.append(0)CRLF
93  ->                 elif sL<L:CRLF
94  ->                     comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT)CRLF
95  ->                     instruct.append(1)CRLF
96  ->                 elif sL>L:CRLF
97  ->                     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT)CRLF
98  ->                     instruct.append(-1)CRLF
```

Python file length: 4,711 lines: 131 Ln: 1 Col: 1 Sel: 0 | 0 Windows (CR LF) UTF-8 2019年3月31日 星期日 INS

code (` `)

```
C:\Users\VrainsHacker\Desktop\MLGame-master\arkanoid\ml\ml_play_original_rule_pass.py - Notepad++
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

ml_play.py x ml_play_original_rule_pass.py x
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
```

Python file length: 4,711 lines: 131 Ln: 1 Col: 1 Sel: 0 | 0 Windows (CR LF) UTF-8 2019年3月31日 星期日 INS



Machine Learning –1 with full data



選了哪種model

- KNN (K-Nearest Neighbors)
- 分群(左移、右移、不動)

如何訓練model

frame和command

```
*C:\Users\WrainsHacker\Desktop\trained_data_knn_model.py - Notepad++
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

trained_data_knn_model.py x
9  import pandas as pd #for handling .csv files
10 import numpy as np
11 from sklearn.model_selection import train_test_split
12
13 #load csv dataset
14 customer_data = pd.read_csv("pass_train_data.csv")
15 customer_data.head(10) # show first ten samples of data
16
17 #extract age and annual income as feature
18 #datax=np.array(customer_data.iloc[:,1:2].values) #將ball拆成x.y進array
19 datax=customer_data.iloc[:,0:1].values
20 datay=customer_data.iloc[:,2:3].values
21
22 #x=data.data
23 #y=data.target
24 # split train and test data into 0.8:0.2
25 x_train,x_test,y_train,y_test=train_test_split(datax,datay,test_size=0.2,random_state=0)
26
27 from sklearn.neighbors import KNeighborsClassifier
28 from sklearn.metrics import accuracy_score
29 neigh = KNeighborsClassifier(n_neighbors=1)
30 neigh.fit(x_train,y_train)
31
32 y_knn=neigh.predict(x_test)
33 # calculate accuracy
34 acc=accuracy_score(y_knn,y_test)
35
36 import pickle
37
38 filename="knn_model_0331.sav"
39 pickle.dump(neigh,open(filename,'wb'))
40
41
42 l_model=pickle.load(open(filename,'rb'))
43 yp_l=l_model.predict(x_test)
44 print("acc load: %f" % accuracy_score(yp_l,y_test))
45
```

Python file length: 1,278 lines: 45 Ln: 35 Col: 1 Sel: 0 | 0 Windows (CR LF) UTF-8 INS

觀察與發現

- KNN
- 3 neighbors --> accuracy: 0.929... --> 無法通關
- 1 neighbors --> accuracy: 0.94... --> 通關

code (` _ `)

C:\Users\VrainsHacker\Desktop\MLGame-master\arkanoid\ml\ml_play.py - Notepad++

檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?




ml_play.py

```
68  -> -> -> # 3.4. Send the instruction for this frame to the game process ..... CR LF
69  -> -> -> arr=np.array([scene_info.frame]) CR LF
70  -> -> -> yp_l=load_model.predict(arr.reshape(1,-1)) CR LF
71  -> -> -> if yp_l==1: CR LF
72  -> -> ->     comm.send_instruction(scene_info.frame,GameInstruction.CMD_LEFT) CR LF
73  -> -> ->     instruct.append(-1) CR LF
74  -> -> -> elif yp_l==2: CR LF
75  -> -> ->     comm.send_instruction(scene_info.frame,GameInstruction.CMD_RIGHT) CR LF
76  -> -> ->     instruct.append(1) CR LF
77  -> -> -> else: CR LF
78  -> -> ->     comm.send_instruction(scene_info.frame,GameInstruction.CMD_NONE) CR LF
79  -> -> ->     instruct.append(0) CR LF
80  -> -> -> CR LF
```



Machine Learning –2 with over 10 seconds data



資料的預處理

- 平台的前後座標差算指令
- 球的前後座標差
- Original Rule 的公式



選了哪種model

- KNN (K-Nearest Neighbors)
- 分群(左移、右移、不動)

如何訓練model

球的前後座標、平台的x座標和指令

```
C:\Users\VrainsHacker\Desktop\MLGame-master\test_train_data\second time test\load_and_train_0401_01.py - Notepad++
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

load_and_train_0401_00.py x load_and_train_0401_01.py x load_and_train_0401_02.py x load_and_train_0401_03.py x ml_play_second_time_ml.py x

1  # -*- coding: utf-8 -*- CR LF
2  """ CR LF
3  Created on Mon Mar 4 17:02:22 2019 CR LF
4  CR LF
5  @author: .scream CR LF
6  """ CR LF
7  import pickle CR LF
8  with open("2019-04-01_09-21-38_22second.pickle", "rb") as f: CR LF
9      data_list = pickle.load(f) CR LF
10 CR LF
11 # save each information seperately CR LF
12 Frame = [] CR LF
13 Status = [] CR LF
14 Ballposition = [] CR LF
15 PlatformPosition = [] CR LF
16 Bricks = [] CR LF
17 for i in range(0, len(data_list)): CR LF
18     Frame.append(data_list[i].frame) CR LF
19     Status.append(data_list[i].status) CR LF
20     Ballposition.append(data_list[i].ball) CR LF
21     PlatformPosition.append(data_list[i].platform) CR LF
22     Bricks.append(data_list[i].bricks) CR LF
23 CR LF
24 %% calculate instruction of each frame using platformposition CR LF
25 import numpy as np CR LF
26 PlatX = np.array(PlatformPosition[:, 0][:, np.newaxis]) CR LF
27 PlatX_next = PlatX[1:, :] CR LF
28 instruct = (PlatX_next - PlatX[0:len(PlatX_next), 0][:, np.newaxis]) / 5 CR LF
29 CR LF
30 CR LF
31 # select some features to make x CR LF
32 Ballarray = np.array(Ballposition[:-1]) CR LF
33 CR LF
34 # PlatX_now = np.array(PlatformPosition[1:530, 0][:, np.newaxis]) CR LF
35 Ballarray_now = Ballarray[1:530, :] CR LF
36 Ballarray_previous = Ballarray[0:529, :] CR LF
37 # Diff = (Ballarray_now - Ballarray_previous) CR LF

Python file length: 2076 lines: 72 ln: 1 col: 1 sel: 010 Windows (CR LF) UTF-8 INS
```


如何訓練model

```
C:\Users\VrainsHacker\Desktop\MLGame-master\test_train_data\second time test\load_and_train_0401_01.py - Notepad++
檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?

load_and_train_0401_00.py x load_and_train_0401_01.py x load_and_train_0401_02.py x load_and_train_0401_03.py x ml_play_second_time_ml.py x
37 #Diff=(Ballarray_now-Ballarray_previous) \r\n
38 #Slope=(Diff[:,1]/Diff[:,0])[:,np.newaxis] \r\n
39 x=np.hstack((Ballarray_now,Ballarray_previous,PlatX[1:530,0][:,np.newaxis])) \r\n
40 #.select.intructions.as.y \r\n
41 y=instruct[1:530,:] \r\n
42 \r\n
43 from sklearn.model_selection import train_test_split \r\n
44 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0) \r\n
45 \r\n
46 ###.train.your.model.here \r\n
47 #from.xxx.import.ooo_model \r\n
48 from sklearn.neighbors import KNeighborsClassifier \r\n
49 #ooo=ooo_model() \r\n
50 neigh = KNeighborsClassifier(n_neighbors=1) \r\n
51 #ooo.fit(x_train,y_train) \r\n
52 neigh.fit(x_train,y_train) \r\n
53 #ooo.predict(x_test) \r\n
54 y_knn=neigh.predict(x_test) \r\n
55 #.check.the.acc.to.see.how.well.you've.trained.the.model \r\n
56 #acc=? \r\n
57 from sklearn.metrics import accuracy_score \r\n
58 acc=accuracy_score(y_knn,y_test) \r\n
59 \r\n
60 \r\n
61 ###.save.model \r\n
62 import pickle \r\n
63 \r\n
64 #filename="ooo_example0401.sav" \r\n
65 filename="neigh_knn_model_0401_n1_02.sav" \r\n
66 pickle.dump(neigh,open(filename,'wb')) \r\n
67 \r\n
68 #.load.model \r\n
69 l_model=pickle.load(open(filename,'rb')) \r\n
70 yp_l=l_model.predict(x_test) \r\n
71 print("acc.load:%f"%accuracy_score(yp_l,y_test)) \r\n
72 .....
```

Python file | length: 2.076 | lines: 72 | Ln: 1 | Col: 1 | Sel: 0 | 0 | Windows (CR LF) | UTF-8 | INS

觀察與發現

- knn
- n_neighbors=3
- `x=np.hstack((Ballarray, PlatX[0:-1,0][:,np.newaxis]))`
- accuracy= 0.8411214953271028
- frame 44
- brick 21
- knn
- n_neighbors=1
- `x=np.hstack((Ballarray, PlatX[0:-1,0][:,np.newaxis]))`
- accuracy= 0.8411214953271028
- frame 44
- brick 21

觀察與發現

- knn
- n_neighbors=1
- x=np.hstack((Ballarray_now,Ballarray_previous,PlatX[1:530,0][:,np.newaxis]))
- accuracy= 0.9245283018867925
- frame 666
- brick 8
- knn
- n_neighbors=1
- x=np.hstack((Ballarray_now,Ballarray_previous,PlatX[1:530,0][:,np.newaxis],L))
- accuracy= 0.9433962264150944
- frame 44
- brick 21
- knn
- n_neighbors=1
- x=customer_data.iloc[:,0:6].values
- y=customer_data.iloc[:,6:7].values
- accuracy= 0.9245283018867925
- frame 534
- brick 10

觀察與發現

- knn
- n_neighbors=1
- `x=np.hstack((Ballarray_now,PlatX[1:530,0][:,np.newaxis],L,Slope,Diff,L2))`
- accuracy= 0.9622641509433962
- frame 448
- brick 11
- knn
- n_neighbors=1
- `x=np.hstack((Ballarray_now,PlatX[1:530,0][:,np.newaxis],L,Slope,Diff,L2, Ballarray_previous))`
- accuracy= 0.9528301886792453
- frame 534
- brick 10

code (` `)

*C:\Users\VrainsHacker\Desktop\MLGame-master\arkanoid\ml\ml_play_second_time_ml.py - Notepad++

檔案(F) 編輯(E) 搜尋(S) 檢視(V) 編碼(N) 語言(L) 設定(T) 工具(O) 巨集(M) 執行(R) 外掛(P) 視窗(W) ?



load_and_train_0401_00.py load_and_train_0401_01.py load_and_train_0401_02.py load_and_train_0401_03.py ml_play_second_time_ml.py

```
66  -> -> # 3.3. Put the code here to handle the scene information CRLF
67  -> -> a=c CRLF
68  -> -> b=d CRLF
69  -> -> c=scene_info.ball[0] CRLF
70  -> -> d=scene_info.ball[1] CRLF
71  -> -> K= ((200-c)*(d-b)/(c-a))-(400-d) CRLF
72  -> -> #KK.append(K) CRLF
73  -> -> #inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], scene_info.platform[0]]) CRLF
74  -> -> inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], a, b, scene_info.platform[0]]) CRLF
75  -> -> #inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], a, b, scene_info.platform[0], K]) CRLF
76  -> -> #inp_temp=np.array([scene_info.ball[0], scene_info.ball[1], a, b, K]) CRLF
77  -> -> input=inp_temp[np.newaxis, :] CRLF
78  CRLF
79  -> -> # 3.4. Send the instruction for this frame to the game process..... CRLF
80  -> -> if load_model.predict(input)==1: CRLF
81  -> ->     comm.send_instruction(scene_info.frame, GameInstruction.CMD_RIGHT) CRLF
82  -> ->     instruct.append(1) CRLF
83  -> -> elif load_model.predict(input)==-1: CRLF
84  -> ->     comm.send_instruction(scene_info.frame, GameInstruction.CMD_LEFT) CRLF
85  -> ->     instruct.append(-1) CRLF
86  -> -> else: CRLF
87  -> ->     comm.send_instruction(scene_info.frame, GameInstruction.CMD_NONE) CRLF
88  -> ->     instruct.append(0) CRLF
```

結論

- ➡ 1.規則寫得好的話，就不用機器學習了
- ➡ 2.目標通關，遊戲設定不變 --> 可用frame index
 - ➡ 平台移動-->連續的frame對應相同的command
- ➡ 3.感覺少量資料訓練不出能通關的，因為是預測分群不會自我學習，或是沒找到關鍵特徵，或是要用其他model
- ➡ 4. accuracy高，實際結果不一定較好



Thanks for your listening.