Trường Đại học Khoa học tự nhiên
Thành phố Hồ Chí Minh

Khoa Công nghệ thông tin

BÁO CÁO

DỰ ĐOÁN NGƯỜI CHƠI CHIẾN THẮNG TRONG GAME PUBG

Môn: Khoa học dữ liệu

Giảng viên: Trần Trung Kiên

Nhóm 1: 1512416 – Nguyễn Tất Nam Phương
1512473 - Trương Ngọc Tài

Tp. Hồ Chí Minh, 8/1/2018.

Nội dung

I) Đặt vấn đề

a) Câu hỏi.

"Ai sẽ là người chiến thắng?"

b) Trả lời.

Nếu trả lời được câu hỏi này:

- Dự đoán trước được người chiến thắng trong game.
- Trường hợp là người chơi: ta có thể biết trước được khả năng thắng/thua của đối thủ để có cách đối phó.
- Đối với 1 tổ chức như Công ty Game hay các đội nhóm chơi game: có thể dựa vào dữ liệu dự đoán này để tiến hành sắp xếp/chọn lọc/cải thiện sức mạnh của đội.

c) Dữ liệu.

Dữ liệu được thu thập thông qua API của PUBG cung cấp.

Hoàn toàn hợp pháp.

II) Giải quyết vấn đề

a) Thu thập dữ liệu.

Lấy dữ liệu thông qua API của PUBG.

Yêu cầu: đã có tài khoảng PUBG

Quy trình:

- Khởi tạo một API với token PUBG cung cấp cho mỗi tài khoảng
- Gửi yêu cầu lên server PUBG để lấy danh sách các giải đấu trên thế giới.
- Ứng với mỗi giải đấu: tiến hành lấy tất cả các trận đấu trong giải đấu đó.
- Ứng với mỗi trận đấu: lấy tất cả các người tham gia trong trận đấu đó.
- Ứng với mỗi người tham gia: lấy tất cả các thuộc tính của người chơi đó.

b) Tiền xử lí dữ liệu.

- Chia bộ dữ liệu thành 3 phần tỉ lệ 60:20:20 tương ứng train:validation:test.
- Phân tách cột: deathType [‘alive’, ‘byplayer’, ‘logout’, ‘suicide’] thành các cột ứng với giá trị trong cột.
    - Tương tự như bag-of-word: chuyển thành one-hot vector rồi nối vào vector feature.
- Groupby cột ‘playerId’
- Loại bỏ các cột không cần thiết: playerId, groupId, matchId, name, deathType(loại sau khi đã phân tách)
    - Vậy ban đầu ta có 38 thuộc tính

```
['DBNOs', 'assists', 'boosts', 'damageDealt', 'deathType', 'headshotKills',
 'heals', 'killPlace', 'killPoints', 'killPointsDelta', 'killStreaks', 'kills',
 'lastKillPoints', 'lastWinPoints', 'longestKill', 'mostDamage', 'name',
 'playerId', 'rankPoints', 'revives', 'rideDistance', 'roadKills', 'swimDistance',
 'teamKills', 'timeSurvived', 'vehicleDestroys', 'walkDistance', 'weaponsAcquired',
 'winPlace', 'winPoints', 'winPointsDelta', 'matchDuration', 'matchID', 'matchType',
 'groupId', 'numGroups', 'maxPlace', 'winPlacePerc']
```

    - Sau khi loại bỏ và phân tách còn: 38-1+4-7-1=33 thuộc tính.

```
['DBNOs', 'assists', 'boosts', 'damageDealt', 'headshotKills',
 'heals', 'killPlace', 'killPoints', 'killPointsDelta',
 'killStreaks', 'kills', 'lastKillPoints', 'lastWinPoints',
 'longestKill', 'mostDamage', 'rankPoints', 'revives',
 'rideDistance', 'roadKills', 'swimDistance', 'teamKills',
 'timeSurvived', 'vehicleDestroys', 'walkDistance', 'weaponsAcquired',
 'winPoints', 'winPointsDelta', 'matchDuration', 'winPlacePerc',
 'deathType_alive', 'deathType_byplayer', 'deathType_logout', 'deathType_suicide']
```

- Điền thêm giá trị thiếu: dùng giá trị trung bình trong tập train để điền cho giá trị thiếu.

```
########################################################################
##################### TRUNG BÌNH các field trong tập TRAIN ######################
DBNOs                9.158219e-01
assists              4.578695e-01
boosts               2.354902e+00
damageDealt          1.698090e+02
headshotKills        1.703966e-01
heals                2.514738e+00
killPlace            4.233843e+01
killPoints           0.000000e+00
killPointsDelta      0.000000e+00
killStreaks          5.171716e-01
kills                8.078514e-01
lastKillPoints       0.000000e+00
lastWinPoints        0.000000e+00
```

- Scale dữ liệu bằng standardScaler của sklearn: chuyển đổi dữ liệu có trung bình bằng 0, độ lệch chuẩn là 1.

```
print(train_X)
```

```
[[-1.17942534e+00 -1.19058336e+00 -2.28242388e+00 ...  4.70525910e-01
  -5.47996624e-02 -2.67554772e-01]
 [ 1.51335121e-01  1.09550719e-01 -2.14749389e-01 ... -1.45166769e-01
  -5.47996624e-02  1.02805805e-01]
 [-1.77777682e-01 -3.23827307e-01  2.30374982e-01 ...  1.44572590e-02
  -5.47996624e-02 -2.67554772e-01]
 ...
 [-6.64292261e-01 -3.23827307e-01  1.72939579e-01 ...  6.00641241e-02
  -5.47996624e-02  4.73166382e-01]
 [ 3.26524992e-02 -7.31712507e-01 -1.90054366e-03 ...  1.08353746e-01
  -5.47996624e-02 -2.67554772e-01]
 [ 3.65973906e-01 -1.50476096e-01 -7.31668012e-01 ...  4.70525910e-01
  -5.47996624e-02 -2.67554772e-01]]
```

- Gắn nhãn cho dữ liệu: winPlacePerc >= 0.5 thì label=1 còn lại là 0.

Kích thước bộ dữ liệu:

```
######################## Kích thước bộ dữ liệu ########################
Train_X: (334, 32)
Validation_X: (111, 32)
Test_X: (112, 32)
########################################################################
```

Train chứ 334 entity.
Validation chứ 111 entity.
Test chứ 112 entity.
Số lượng feature là 32.

c) Xây dựng mô hình để mô hình hóa dữ liệu.

Mô hình 1: Perceptron

- Tham số:
  - Alpha=0.001
  - Max_iter=1
  - randomState=0

Mô hình 2: Multi Layer Perceptron

- Tham số:
  - 2 lớp ẩn: lớp thứ nhất có 50 node, lớp thứ 2 có 25 node ẩn.
  - Activation='tanh'.
  - Solver='lbfgs'
  - Max_iter=1000
  - randomState=0

Mô hình 3: Stochastic Gradient Descent

- Tham số:
  - Alpha=0.001
  - Max_iter=100

Mô hình 4: Logistic Regression

- Tham số:
  - C=1
  - Max_iter=100

d) Huấn luyện mô hình.

Huấn luyện mô hình với dữ liệu đã được tiền xử lí trên và với nhã đã được gắn tương ứng.

Training model

```
def trainingModel(model, train_X, train_y):
    model.fit(train_X, train_y)
```

e) Kiểm thử và chọn mô hình phù hợp.

Tiến hành kiểm thử từng mô hình dữ liệu đã xây dựng trên tập dữ liệu validation.

Kết quả kiểm thử:

```
Model Perceptron:
0.218562874251497
0.32432432432432434
------------------
------------------
Model MLP:
0.0
0.2972972972972973
------------------
------------------
Model SGD:
0.19461077844311378
0.3153153153153153
------------------
------------------
Model Logistic Regression:
0.16467065868263472
0.36036036036036034
------------------
------------------
```

f) Đánh giá mô hình được chọn với dữ liệu thực.

Mô hình tốt nhất là MLP.

Kết quả khi kiểm tra trên tập test:

```
------------------
------------------
Đánh giá model được chọn - MLP!

Độ lỗi trên tập test
0.22321428571428573
------------------
------------------
```

III) Tham khảo.

- Kaggel: https://www.kaggle.com
- PUBG API: https://documentation.playbattlegrounds.com/en/getting-started.html