NBA勝負預測

Group A

植微四 馬詠芝、植微四 王靖曈
心理四 徐聖琁、心理四 李彥廷
生醫電資所博三 王子毅

TABLE OF CONTENTS

研究主題與目標

- 以 NBA 球隊比賽數據和球員資料來預測兩隊比賽的勝負
- 目標:
  - 建置準確的預測模型
  - 了解重要特徵值
  - 部署網頁以呈現成果

資料處理 - 篩選與前處理

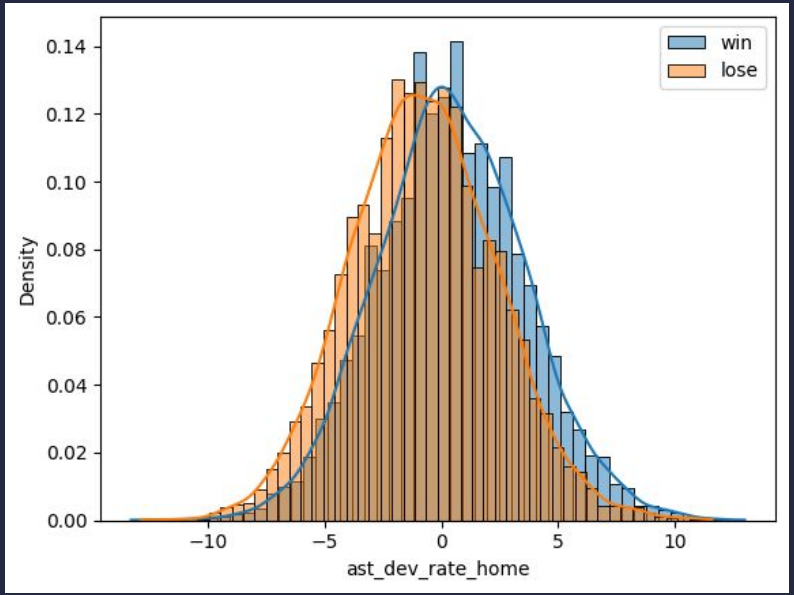
球隊傳統數據:

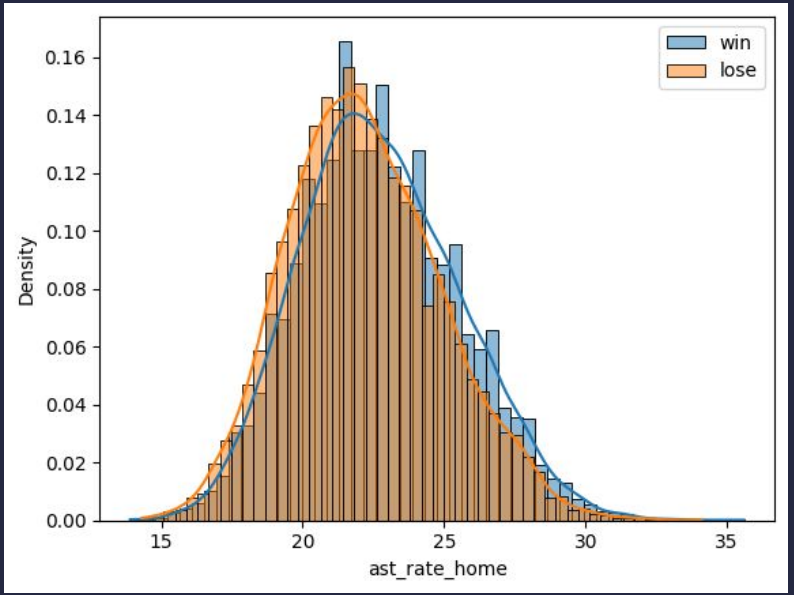
1. 選擇 90-91 賽季後的逐場數據
2. 整理成主場球隊和客場球隊的數據差 (主-客)
3. 以主客隊「過去十場的平均」進行預測

球員相關資料:

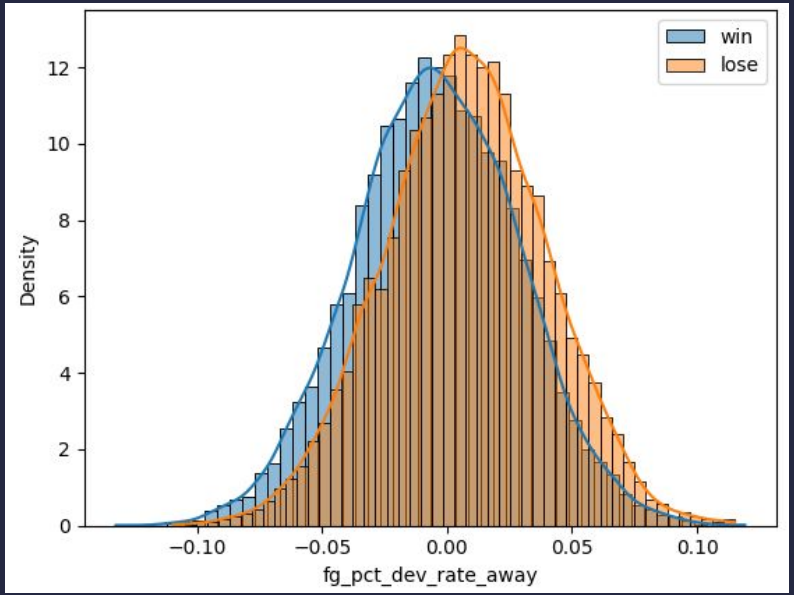
1. 以球隊中「入選全明星賽之球員數」、「年度前三隊球員數」、「MVP 排行榜之球員數」作為球隊陣容實力指標

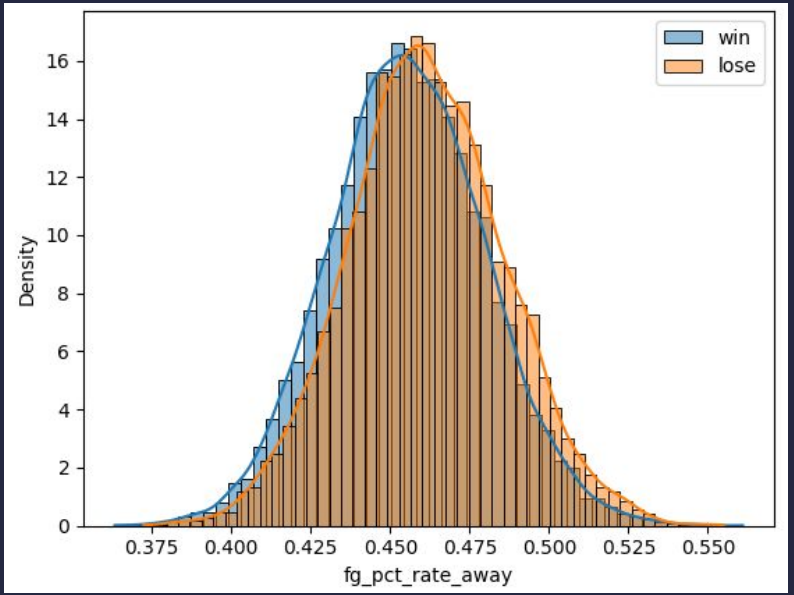
資料處理 - 特徵值分析


win
lose
Density
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−10
−5
0
5
10
ast_dev_rate_home

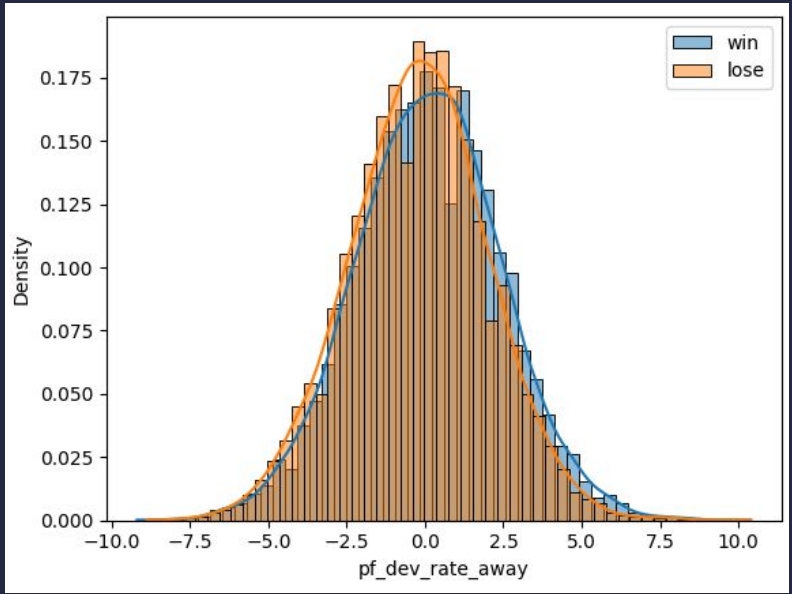

win
lose
Density
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
15
20
25
30
35
ast_rate_home

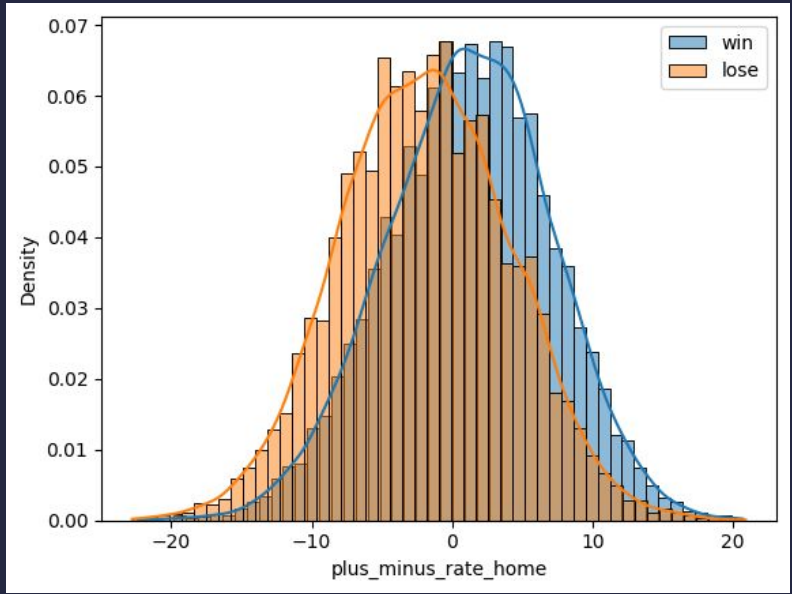
資料處理 - 特徵值分析


win
lose
Density
fg_pct_dev_rate_away
−0.10
−0.05
0.00
0.05
0.10
0
2
4
6
8
10
12


win
lose
Density
fg_pct_rate_away
0.375
0.400
0.425
0.450
0.475
0.500
0.525
0.550
0
2
4
6
8
10
12
14
16

資料處理 - 特徵值分析


win
lose
Density
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
pf_dev_rate_away


win
lose
Density
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
−20
−10
0
10
20
plus_minus_rate_home

Result
NBA Game Predictions Heatmap
Data Columns
ast_dev_rate
dreb_dev_rate
fg3_pct_dev_rate
fg3a_dev_rate
fg3m_dev_rate
fg_pct_dev_rate
fgm_dev_rate
fta_dev_rate
ftm_dev_rate
oreb_dev_rate
reb_dev_rate
ast_rate
blk_rate
dreb_rate
fg3a_rate
fg3m_rate
fga_rate
fgm_rate
ft_pct_rate
fta_rate
ftm_rate
min_rate
oreb_rate
plus_minus_rate
reb_rate
tov_rate
wl_rate
MVP_in_roster_share_2yr
MVP_in_roster_share_3yr
MVP_in_roster_share_1yr
All_nba_2nd_team_players_3yr
All_nba_1st_team_players_3yr
All_nba_3rd_team_players_3yr
All_nba_3rd_team_players_1yr
All_nba_2nd_team_players_1yr
All_nba_1st_team_players_1yr
All_nba_2nd_team_players_2yr
All_nba_3rd_team_players_2yr
All_nba_1st_team_players_2yr
Team_all_stars
0.0
0.2
0.4
0.6
0.8
1.0
Normalized Values

模型建置與調整

- **Pycaret**
- **Auto-sklearrn**
- **TPOT**

PyCaret

Pycaret特色

來自R Caret

Classification and
Regression Training

資料解釋、視覺化

用簡單的指令即獲得
詳細的報告及圖表

開源＆低程式碼

適合素人資料科學家

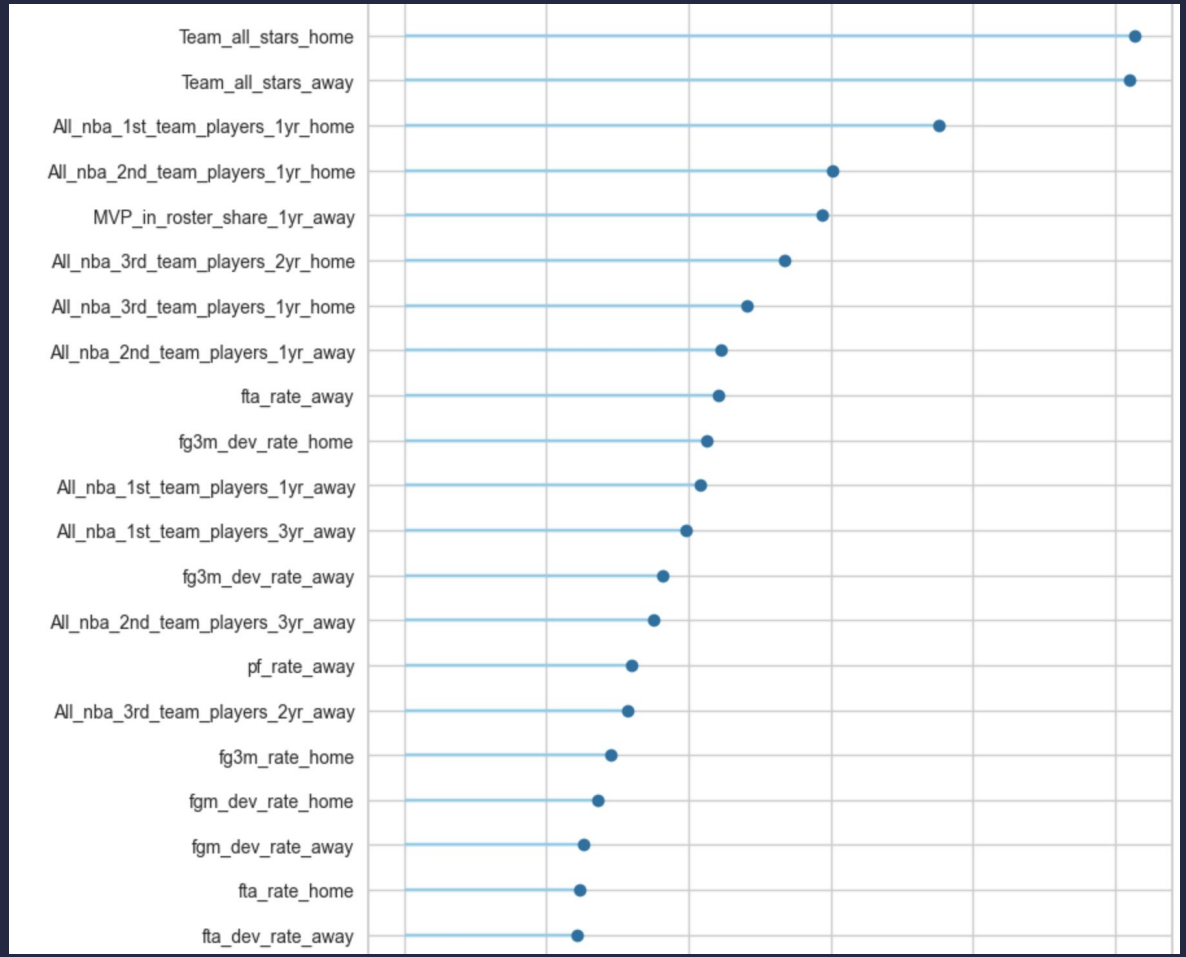
相容性

無痛切換至所有支
援Python的環境

前10名important features

8
入選全明星賽人數
前一年獲選年度前三隊人數
前一年 MVP 排行榜球員數

2
罰球數
三分球命中率



前10名important features

8
入選全明星賽人數
前一年獲選年度前三隊人數
前一年 MVP 排行榜球員數

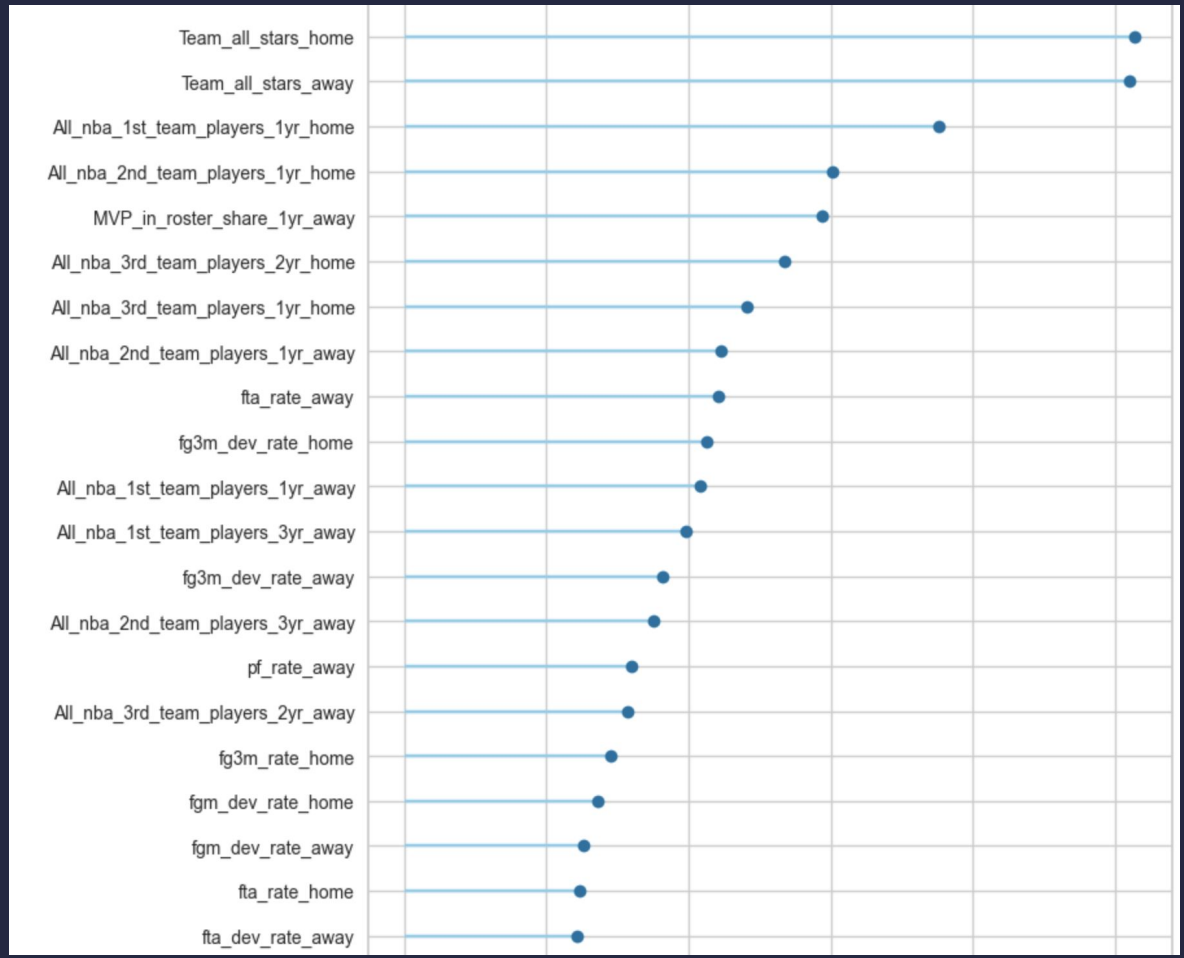
2
罰球數
三分球命中率

LogisticRegression

Training data: 66.8%

Testing data: 69.9%

球員影響力>>球隊影響力



Autosklearn

AutoSklearn

- 基於Sklearn開發
- 自動測試多種模型及超參數
- 最後輸出一個集成模型

AutoSklearn

- 只有球隊資料 (2 hr)

```
automlclassifierV1 Training dataset:  0.6666222340709145
automlclassifierV1 Testing dataset:  0.661103047895507
```

- 加入球員資料、dev (1 hr)

```
automlclassifierV1 Training dataset:  0.7100423778235683
automlclassifierV1 Testing dataset:  0.6793655730057534
```

Autosklearn 最終的模型內容

Rank	Ensemble_weight	Model	Cost
1	0.02	Stochastic Gradient Descent	0.3293029
2	0.32	Gradient_boosting	0.3297694
3	0.42	Linear Discriminant Analysis	0.3308192
4	0.1	Gradient_boosting	0.3354457
5	0.12	Quadratic Discriminant Analysis	0.3433381
6	0.02	Gradient_boosting	0.4008009

TPOT

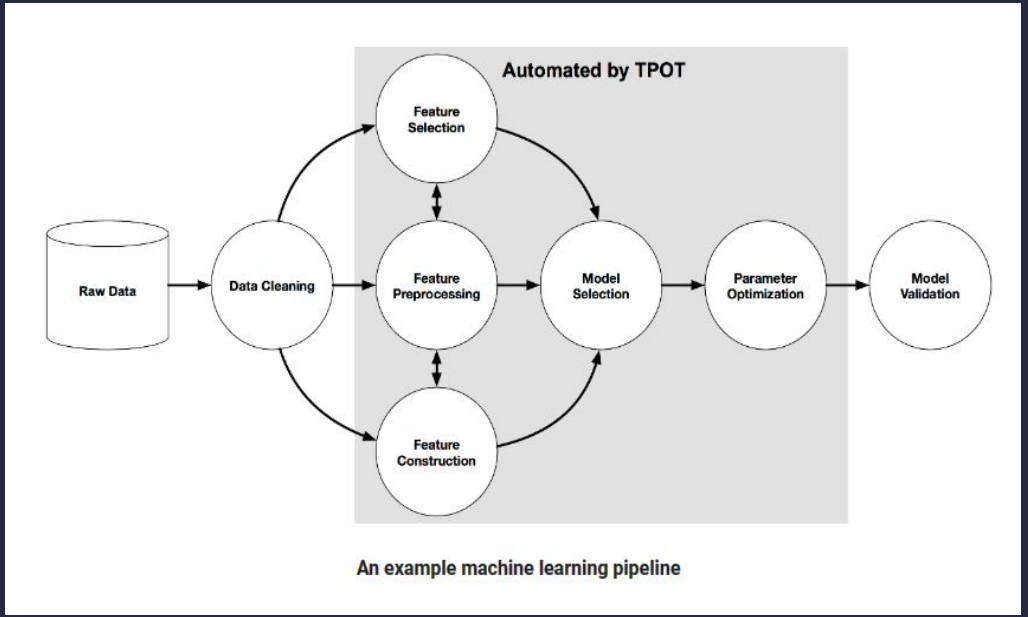
Tree-based Pipeline Optimization Tool

TPOT特色

使用Genetic Programming

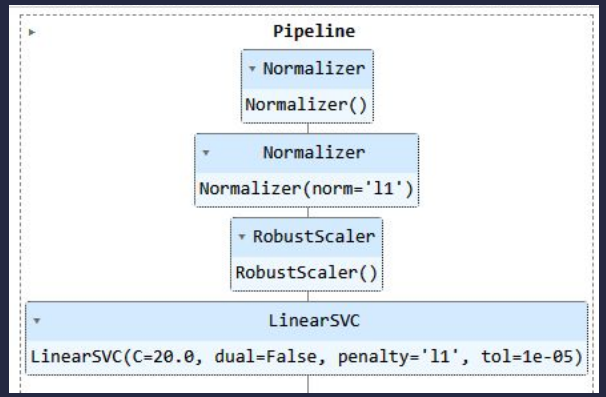
自動產生最佳模型訓練及測試的pipeline並提供程式碼

可設定提早停止訓練的時間，且停止之後可以接續訓練。



An example machine learning pipeline

TPOT




```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import Normalizer, RobustScaler
from sklearn.svm import LinearSVC

# NOTE: Make sure that the outcome column is labeled 'target' in the data file
tpot_data = pd.read_csv('PATH/TO/DATA/FILE', sep='COLUMN_SEPARATOR', dtype=np.float64)
features = tpot_data.drop('target', axis=1)
training_features, testing_features, training_target, testing_target = \
            train_test_split(features, tpot_data['target'], random_state=None)

# Average CV score on the training set was: 0.6783561074176807
exported_pipeline = make_pipeline(
    Normalizer(norm="l2"),
    Normalizer(norm="l1"),
    RobustScaler(),
    LinearSVC(C=20.0, dual=False, loss="squared_hinge", penalty="l1", tol=1e-05)
)

exported_pipeline.fit(training_features, training_target)
results = exported_pipeline.predict(testing_features)
```

Training data: 67.8%

Testing data: 67.3%

Prediction Interface Demo

結論

- 建置之模型準確度:
  - Pycaret: 69.9%
  - TPOT: 67.3%
  - Auto-scikitlearn: 67.9%

  ⇒ 前人表明 NBA Upset rate = 30%，故本研究完全符合文獻
- 球隊的「球員陣容」主宰比賽勝負
- 成果部署於網頁供使用者使用

Thanks for Listening

Q&A