create_gamelog_tensors

May 10, 2020

[13]:
```
import os
import pandas as pd
import numpy as np
from tensorflow.keras.models import load_model
from joblib import load
pd.options.mode.chained_assignment = None  # default='warn'
```

[14]:
```
bat = load_model('../core/models/model_batting.h5')
pitch = load_model('../core/models/model_pitching.h5')
bat_scaler = load('../core/models/batting_scaler.save')
pitch_scaler = load('../core/models/pitching_scaler.save')
```

[15]:
```
gl = pd.read_csv('../core/data/retrosheet/gamelogs/GL2015.csv')
```

[16]:
```
gl
```

[16]:

	visit_team	home_team	visit_score	home_score	game_length_outs	\
0	MIN	DET	0	4	51	
1	CLE	HOU	0	2	51	
2	CHW	KCR	1	10	51	
3	TOR	NYY	6	1	54	
4	TEX	OAK	0	8	51	
...	...	...	...	...	...	
2423	CHC	MIL	3	1	54	
2424	WSN	NYM	0	1	51	
2425	FLA	PHI	2	7	51	
2426	CIN	PIT	0	4	51	
2427	COL	SFG	7	3	54	

	night_game	park_id	visit_manager_id	home_manager_id	visit_sp_id	...	\
0	0	DET05	molip001	ausmb001	hughp001	...	
1	1	HOU03	frant001	hinca001	klubc001	...	
2	0	KAN06	ventr001	yoste001	samaj001	...	
3	0	NYC21	gibbj001	giraj001	hutcd001	...	
4	1	OAK01	banij001	melvb001	gally001	...	
...	...	...	...	...	...	...	
2423	0	MIL06	maddj801	counc001	hared001	...	

```
2424           0   NYC20         willm003        collt801    roart001  ...
2425           0   PHI13         jennd801        mackp101    conla001  ...
2426           0   PIT08         pricb801        hurdc001    smitj004  ...
2427           0   SFO03         weisw001        bochb002    bergc001  ...

     home_player_4_id home_player_5_id home_player_6_id home_player_7_id  \
0           martv001         martj006         cespy001         castn001
1           gatte001         cartc002         castj006         lowrj001
2           hosme001         morak001         gorda001         riosa002
3           teixm001         mccab002         headc001         rodra001
4           butlb003         davii001         lawrb002         vogts001
...              ...              ...              ...              ...
2423        davik003         santd002         pereh001         seguj002
2424        cespy001         dudal001         darnt001         confm001
2425        ruf-d001         franj004         blana001         krate001
2426        walkn001         marts002         alvap001         cervf001
2427        poseb001         parkj002         willm008         noonn001

     home_player_8_id home_player_9_id  year month day home_win
0           avila001         iglej001  2015     4   6        1
1           rasmc001         marij002  2015     4   6        1
2           peres002         infao001  2015     4   6        1
3           drews001         gregd001  2015     4   6        0
4           semim001         sogae001  2015     4   6        1
...              ...              ...   ...   ...  ..      ...
2423        maldm001         lopej004  2015    10   4        0
2424        tejar001         degrj001  2015    10   4        1
2425        ruppc001         buchd001  2015    10   4        1
2426        mercj002         happj001  2015    10   4        1
2427        willj005         cainm001  2015    10   4        0

[2428 rows x 33 columns]
```

[17]:
```
columns = {
    'batting': [],
    'pitching': []
}
```

[18]:
```
batters = pd.read_csv('../core/output/batters.csv')
batter_years = pd.read_csv('../core/output/batting.csv')
batters_not_counted = list(batter_years[~batter_years['retroID']
                                        .isin(batters['retroID'])]['retroID'].
 ↪values)
pitchers = pd.read_csv('../core/output/pitchers.csv')
pitcher_years = pd.read_csv('../core/output/pitching.csv')
bat_scaler = load('../core/models/batting_scaler.save')
pitch_scaler = load('../core/models/pitching_scaler.save')
```

```
scalers = {
    'batting': bat_scaler,
    'pitching': pitch_scaler
}
career_features = {
    'batting': [
        'G', 'AB', 'PA', 'R', 'H', '1B', '2B', '3B',
        'HR', 'RBI', 'SB', 'CS', 'BB', 'SO', 'IBB',
        'HBP', 'SH', 'SF', 'GIDP'
    ],
    'pitching': [
        'CG', 'SHO', 'H', 'ER', 'HR', 'BB', 'SO',
        'BAOpp', 'ERA', 'IBB', 'WP', 'HBP', 'BK',
        'BFP', 'GF', 'R', 'SH', 'SF', 'GIDP'
    ]
}
unwanted_features = {
    'batting': ['retroID', 'G', 'AB', '1B', 'RBI', 'wOBA', 'Batting'],
    'pitching': ['IPouts', 'BFP', 'R', 'Pitching']
}
players = {
    'batting': {
        'players': batters,
        'years': batter_years
    },
    'pitching': {
        'players': pitchers,
        'years': pitcher_years
    }
}
```

[19]:
```
def to_tensor_input(scaler, player, label):
    scalers[label] = scaler
    return scaler.transform(player.values.reshape(-1, player.shape[0]))[0]


def convert_single_player(retro_id, year, player_type_label):
    scaler = scalers[player_type_label]
    if retro_id in batters_not_counted:
        return np.zeros(shape=(1, 30))
    player_table = players[player_type_label]['players']
    player_so_far_table = players[player_type_label]['years']
    player = player_table[player_table['retroID'] == retro_id]
    player_so_far = player_so_far_table[(player_so_far_table['retroID'] ==␣
 ↪retro_id)
                                        & (player_so_far_table['yearID'] <=␣
 ↪year)]
```

```
    if not player.size | player_so_far.size:
        print('Handled: {}'.format(retro_id))
        return np.zeros(shape=(1, 30))
    player_so_far = player_so_far.groupby('retroID').sum()
    features = career_features[player_type_label]
    try:
        for column in player[features]:
            player.iloc[0][column] = player_so_far.iloc[0][column]
    except:
        print(retro_id)
    player_columns_to_drop = unwanted_features[player_type_label]
    player = player.drop(columns=player_columns_to_drop)
    if not len(list(columns[player_type_label])):
        columns[player_type_label] = player.columns
    return to_tensor_input(scaler, player.T, player_type_label)


def get_batter_as_tensor_input(batter, year):
    scaler = scalers['batting']
    player = batters[batters['retroID'] == batter]
    player_so_far = batter_years[(batter_years['retroID'] == batter)
                                 & (batter_years['yearID'] <= year)]
    player_so_far = player_so_far.groupby('retroID').sum()
    features = ['G', 'AB', 'PA', 'R', 'H', '1B', '2B', '3B',
                'HR', 'RBI', 'SB', 'CS', 'BB', 'SO', 'IBB',
                'HBP', 'SH', 'SF', 'GIDP']
    for column in player[features]:
        player.iloc[0][column] = player_so_far.iloc[0][column]
    player_columns_to_drop = ['retroID', 'wOBA', 'Batting']
    player = player.drop(columns=player_columns_to_drop)
    return to_tensor_input(scaler, player, 'batting')
```

[20]:
```
convert_single_player('bettm001', 2015, 'batting')
```

[20]:
```
array([0.42623   , 0.3       , 0.        , 0.        , 0.        ,
       0.        , 1.        , 0.        , 0.        , 0.        ,
       0.        , 0.16129032, 0.2288002 , 0.2671024 , 0.22673872,
       0.30697051, 0.13612565, 0.1824147 , 0.08961593, 0.07462687,
       0.14503518, 0.17866769, 0.03633721, 0.06666667, 0.01486989,
       0.25      , 0.10379747, 0.        , 0.21133094, 0.26473988])
```

[21]:
```
gl.iloc[43]
```

[21]:
```
visit_team                SFG
home_team                 SDP
visit_score                 1
home_score                  0
```

```
game_length_outs           72
night_game                  0
park_id                 SAN02
visit_manager_id     bochb002
home_manager_id      blacb001
visit_sp_id          hudst001
home_sp_id           kenni001
visit_player_1_id    aokin001
visit_player_2_id    panij002
visit_player_3_id    pagaa001
visit_player_4_id    poseb001
visit_player_5_id    crawb001
visit_player_6_id    mcgec001
visit_player_7_id    blang001
visit_player_8_id    ariaj001
visit_player_9_id    hudst001
home_player_1_id     myerw001
home_player_2_id     norrd001
home_player_3_id     kempm001
home_player_4_id     uptoj001
home_player_5_id     middw001
home_player_6_id     alony001
home_player_7_id     gyorj001
home_player_8_id     amara001
home_player_9_id     kenni001
year                    2015
month                      4
day                        9
home_win                   0
Name: 43, dtype: object
```

[22]:
```
v1 = gl.iloc[0]['visit_player_1_id']
```

[23]:
```
v1
```

[23]:
```
'santd001'
```

[24]:
```
visit_id = []
home_id = []
```

[25]:
```
for i in range(1, 10):
    visit_id.append(gl.iloc[43]['visit_player_{}_id'.format(i)])
    home_id.append(gl.iloc[43]['home_player_{}_id'.format(i)])
```

[26]:
```
visit_id
```

[26]:
```
['aokin001',
 'panij002',
 'pagaa001',
 'poseb001',
 'crawb001',
 'mcgec001',
 'blang001',
 'ariaj001',
 'hudst001']
```

[27]:
```
gl.iloc[0]['year']
```

[27]:
```
2015
```

[28]:
```
visit = []
home = []
year = gl.iloc[43]['year']
for index in range(0, 9):
    vrid = visit_id[index]
#     vpos = 'pitching' if vrid == gl.iloc[0]['visit_sp_id'] else 'batting'
    vplayer = convert_single_player(vrid, year, 'batting')
    visit.append(vplayer)
    hrid = home_id[index]
#     hpos = 'pitching' if hrid == gl.iloc[0]['home_sp_id'] else 'batting'
    hplayer = convert_single_player(hrid, year, 'batting')
    home.append(hplayer)
```

[29]:
```
visit[0].shape
```

[29]:
```
(30,)
```

[30]:
```
home[0].shape
```

[30]:
```
(30,)
```

[31]:
```
gl.columns
```

[31]:
```
Index(['visit_team', 'home_team', 'visit_score', 'home_score',
       'game_length_outs', 'night_game', 'park_id', 'visit_manager_id',
       'home_manager_id', 'visit_sp_id', 'home_sp_id', 'visit_player_1_id',
       'visit_player_2_id', 'visit_player_3_id', 'visit_player_4_id',
       'visit_player_5_id', 'visit_player_6_id', 'visit_player_7_id',
       'visit_player_8_id', 'visit_player_9_id', 'home_player_1_id',
       'home_player_2_id', 'home_player_3_id', 'home_player_4_id',
       'home_player_5_id', 'home_player_6_id', 'home_player_7_id',
       'home_player_8_id', 'home_player_9_id', 'year', 'month', 'day',
       'home_win'],
```

```
      dtype='object')
```

[32]:
```
batters = visit + home
```

[33]:
```
dfb = pd.DataFrame(batters, columns=columns['batting'])
```

[34]:
```
dfb
```

[34]:

	weight	height	pos_1B	pos_2B	pos_3B	pos_C	pos_OF	pos_P	pos_SS	\
0	0.426230	0.30	0.0	0.0	0.0	0.0	1.0	0.0	0.0	
1	0.508197	0.50	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
2	0.508197	0.55	0.0	0.0	0.0	0.0	1.0	0.0	0.0	
3	0.549180	0.50	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
4	0.618852	0.55	0.0	0.0	0.0	0.0	0.0	0.0	1.0	
5	0.590164	0.50	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
6	0.454918	0.35	0.0	0.0	0.0	0.0	1.0	0.0	0.0	
7	0.446721	0.50	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
8	0.405738	0.50	0.0	0.0	0.0	0.0	0.0	1.0	0.0	
9	0.528689	0.60	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
10	0.651639	0.45	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
11	0.610656	0.65	0.0	0.0	0.0	0.0	1.0	0.0	0.0	
12	0.569672	0.50	0.0	0.0	0.0	0.0	1.0	0.0	0.0	
13	0.590164	0.60	0.0	0.0	1.0	0.0	0.0	0.0	0.0	
14	0.631148	0.50	1.0	0.0	0.0	0.0	0.0	0.0	0.0	
15	0.569672	0.35	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
16	0.344262	0.15	0.0	1.0	0.0	0.0	0.0	0.0	0.0	
17	0.528689	0.45	0.0	0.0	0.0	0.0	0.0	1.0	0.0	

	bats_L	...	BB	SO	IBB	HBP	SH	SF	\
0	1.0	...	0.091478	0.099345	0.004360	0.168421	0.111524	0.117188	
1	1.0	...	0.085614	0.097420	0.020349	0.073684	0.048327	0.203125	
2	0.0	...	0.124707	0.245668	0.026163	0.014035	0.078067	0.265625	
3	0.0	...	0.189210	0.244128	0.090116	0.147368	0.003717	0.398438	
4	1.0	...	0.155590	0.360801	0.091570	0.129825	0.022305	0.367188	
5	0.0	...	0.097342	0.199076	0.020349	0.028070	0.000000	0.242188	
6	1.0	...	0.141908	0.254524	0.030523	0.052632	0.096654	0.117188	
7	0.0	...	0.014464	0.057759	0.010174	0.028070	0.044610	0.070312	
8	0.0	...	0.010164	0.073161	0.000000	0.007018	0.249071	0.015625	
9	0.0	...	0.122361	0.322680	0.020349	0.045614	0.003717	0.156250	
10	0.0	...	0.076231	0.208317	0.014535	0.066667	0.007435	0.093750	
11	0.0	...	0.196638	0.616095	0.098837	0.091228	0.003717	0.562500	
12	0.0	...	0.284988	0.692337	0.071221	0.235088	0.011152	0.429688	
13	0.0	...	0.025020	0.125144	0.005814	0.031579	0.003717	0.078125	
14	1.0	...	0.143081	0.249519	0.042151	0.056140	0.003717	0.218750	
15	0.0	...	0.091087	0.243358	0.007267	0.059649	0.000000	0.171875	
16	1.0	...	0.042611	0.113593	0.018895	0.014035	0.085502	0.117188	
17	0.0	...	0.012901	0.059299	0.000000	0.003509	0.156134	0.015625	

```
        GIDP   NL      wRC+       WAR
0   0.124051  1.0  0.184353  0.105202
1   0.129114  1.0  0.177158  0.103468
2   0.136709  1.0  0.181655  0.157803
3   0.374684  1.0  0.205036  0.354335
4   0.235443  1.0  0.173561  0.172254
5   0.291139  1.0  0.171763  0.072832
6   0.088608  1.0  0.173561  0.101734
7   0.060759  1.0  0.158273  0.051445
8   0.030380  1.0  0.095324  0.058382
9   0.179747  1.0  0.186151  0.104624
10  0.106329  1.0  0.171763  0.105202
11  0.453165  1.0  0.198741  0.206358
12  0.313924  1.0  0.197842  0.262428
13  0.081013  1.0  0.158273  0.055491
14  0.237975  1.0  0.181655  0.080925
15  0.184810  1.0  0.179856  0.100000
16  0.081013  1.0  0.147482  0.036416
17  0.007595  1.0  0.097122  0.055491

[18 rows x 30 columns]
```

[35]:
```
btensor = [dfb, gl.iloc[43]['home_win']]
```

[36]:
```
# btensor
```

[37]:
```
gl.shape
```

[37]:
```
(2428, 33)
```

[38]:
```
gl.shape[0]
```

[38]:
```
2428
```

[39]:
```
players['batting']['players']['retroID'].str.contains('aardd001').sum() == 1
```

[39]:
```
True
```

Modular script to handle all gamelogs

[40]:
```
cols = list(columns['batting'].values) + ['Result']
for year in range(1919, 2020):
    df = pd.DataFrame()
    print('{}'.format(year))
#     gl = pd.read_csv('../core/data/retrosheet/gamelogs/GL{}.csv'.format(year))
#     for index in range(0, gl.shape[0]):
```

```
#         visit_id = []
#         home_id = []
#         for i in range(1, 10):
#             visit_id.append(gl.iloc[index]['visit_player_{}_id'.format(i)])
#             home_id.append(gl.iloc[index]['home_player_{}_id'.format(i)])
#         visit = []
#         home = []
#         for i in range(0, 9):
#             vrid = visit_id[i]
#             vplayer = convert_single_player(vrid, year, 'batting')
#             visit.append(vplayer)
#             hrid = home_id[i]
#             hplayer = convert_single_player(hrid, year, 'batting')
#             home.append(hplayer)
#         batters = list(np.append(np.array(visit + home).flatten(), gl.
 ↪iloc[index]['home_win']))
#         try:
#             bat_df = pd.DataFrame(batters)
#         except:
#             print('{0}\n{1}'.format(vrid))
#         df = df.append(bat_df.T)

#     if not os.path.exists('../core/tensors/games/'):
#         os.mkdir('../core/tensors/games/')
#     df.to_csv('../core/tensors/games/{0}.csv'.format(str(year)), index=False,␣
 ↪header=None)
```

1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986

```
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
```

[ ]: