research

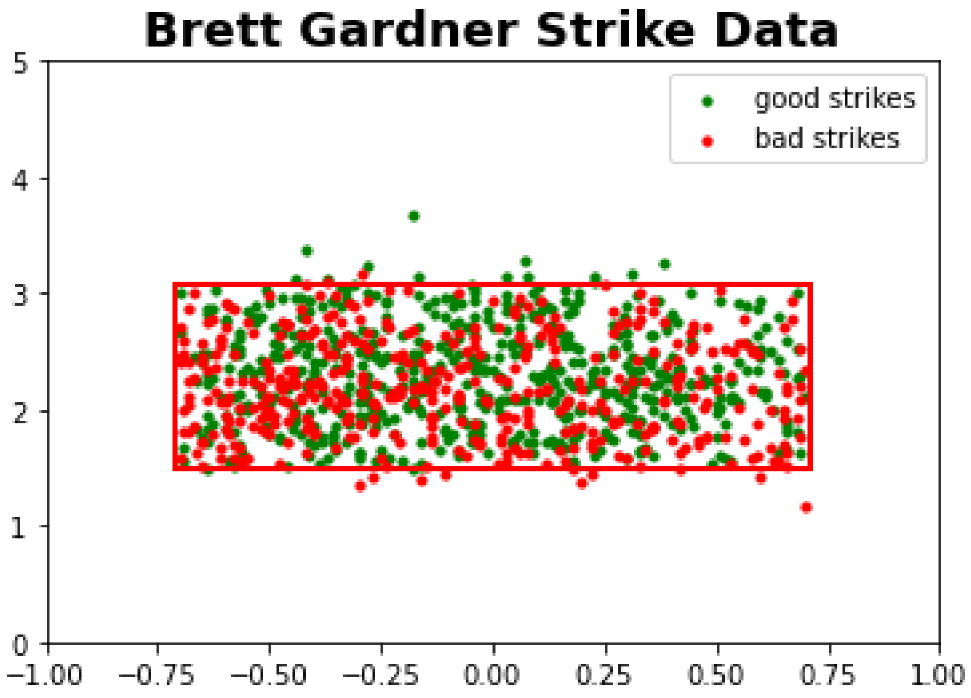
September 21, 2021

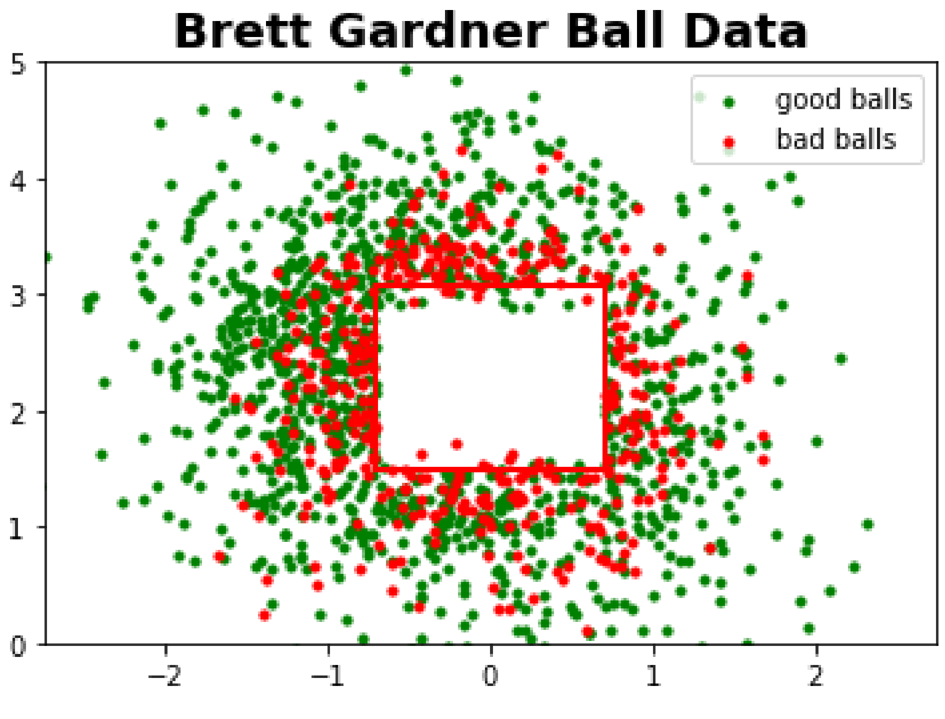
```
[45]: #https://baseballsavant.mlb.com/csv-docs#plate_x
from pybaseball import statcast
from pybaseball import playerid_lookup
from pybaseball import statcast_pitcher, pitching_stats
from pybaseball import statcast_batter, batting_stats_range, batting_stats
from pybaseball import team_ids, teams, team_batting, team_batting_bref
from pybaseball import cache, standings, get_splits
import pandas as pd
import os
import matplotlib
from matplotlib import pyplot as plt
from matplotlib.patches import Rectangle
import matplotlib.patches as mpatchess
cache.enable()

def Average(lst):
    return(sum(lst)/len(lst))

start_date = '2019-01-01' #yy/mm/dd
end_date = '2019-12-31'
data = statcast(start_date, end_date)
#print(data.head())

print("=============================== PITCH DATA NYY␣
 ↪=============================== \n")
dataNYY = team_batting_bref('NYY', 2019)
players = dataNYY['Name'].to_list()
listPLayers = []     #### all NYY players
for playerName in players:
    name = playerName.split()
    listPLayers.append(name)

#### Stores players' ID in dict... Omits 2 players - no data
playerNums = {}
for player in listPLayers:
    playerName = player[0] + " " + player[1]
    playerLookUp = playerid_lookup(player[1], player[0])
    playerNum = playerLookUp['key_mlbam'].to_list()
    if len(playerNum) != 0:
        playerNums[playerName] = playerNum[0]   #### {'key': 'value'}

#print(data.columns.get_loc('plate_x'))     #### finds index
#print(data.columns.get_loc('plate_z'))
#print(data.columns.get_loc('sz_top'))
#print(data.columns.get_loc('sz_bot'))
#print("Balls: ",data.columns.get_loc('balls'))  #24
#print("Strikes: ",data.columns.get_loc('strikes'))  #25

index = data.index
num_rows = len(index)

YankAtBats = []     #### all yank pitches
xPosList = []       #### all x positions
zPosList = []       #### all z positions
DescList = []       #### all descriptions of pitches
topSZ = []          #### top of strike zone - predetermined
botSZ = []          #### bottom of strike zone - predetermined
pType = []          #### pitch type
Names = []          #### batter name
Count_3_2 = []
i = 0
gardyT = []
gardyB = []
for x in range(num_rows):
    ID = (data.iloc[x]).iloc[6]
    name = [key for key, v in playerNums.items() if v == ID]
    if name != [] and name[0] in playerNums:
        Names.append(name[0])
        xPosList.append((data.iloc[x]).iloc[29])
        zPosList.append((data.iloc[x]).iloc[30])
        pType.append((data.iloc[x]).iloc[0])
        DescList.append((data.iloc[x]).iloc[9])
        topSZ.append((data.iloc[x]).iloc[50])
        botSZ.append((data.iloc[x]).iloc[51])
        if ((data.iloc[x]).iloc[24]) == 3 and ((data.iloc[x]).iloc[25]) == 2:␣
 ↪#### pulls out all 3-2 counts
            Count_3_2.append(1)
        else:
            Count_3_2.append(0)
        if name[0] == "Brett Gardner":
            gardyT.append((data.iloc[x]).iloc[50])
            gardyB.append((data.iloc[x]).iloc[51])
        atBat = [Names[i], xPosList[i], zPosList[i], pType[i], DescList[i]]
        YankAtBats.append(atBat)
        i += 1

#### Classifications
classification = []
j = 0
for yankPitch in YankAtBats:
    ## strike
    if (-0.71 < xPosList[j] < 0.71) and (botSZ[j] < zPosList[j] < topSZ[j]):
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
            classification.append("strike, good") # swung
        else:
            classification.append("strike, bad")  # didn't swing
    ## not strike
    else:
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
            classification.append("ball, bad")    # swung
        else:
            classification.append("ball, good")   # didn't swing
    YankAtBats[j].append(classification[j])
    j += 1

#### New data frame of all NYY pitches (size = 28512, last 53 bad)
newdf = pd.DataFrame(YankAtBats[0:len(YankAtBats)-53], columns = ['Name',␣
 ↪'xPos', 'zPos', 'Pitch Type', 'Given Description', 'Classification'])
print("FA - Fastball, CU - Curveball, FT - Two-seam Fastball, CH - Changeup,␣
 ↪\nFC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,\nFF - Four-seam␣
 ↪Fastball \n")
print(newdf,"\n")
```

```
This is a large query, it may take a moment to complete

  0%|          | 0/225 [00:00<?, ?it/s]

Skipping offseason dates
Skipping offseason dates

100%|      | 225/225 [01:56<00:00,  1.93it/s]

=============================== PITCH DATA NYY ===============================


FA - Fastball, CU - Curveball, FT - Two-seam Fastball, CH - Changeup,
FC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,
FF - Four-seam Fastball

                    Name  xPos  zPos Pitch Type Given Description  \
0         Gleyber Torres -0.49  3.21         FT     hit_into_play
1         Gleyber Torres  0.39  1.85         SL     called_strike
2            Aaron Judge  0.31  2.99         FF   swinging_strike
3            Aaron Judge  0.77  1.03         FC              ball
4            Aaron Judge  0.83  1.81         FC     called_strike
…                  …    …    …         …                …
28454  Edwin Encarnacion -0.58  2.31         SL     hit_into_play
28455  Edwin Encarnacion -1.11  4.14         FF              ball
28456  Edwin Encarnacion -0.39  1.81         SL   swinging_strike
28457  Edwin Encarnacion  0.30  1.09         CH              ball
28458  Edwin Encarnacion  0.92  3.22         FT     called_strike

      Classification
0          ball, bad
1        strike, bad
2       strike, good
3         ball, good
4         ball, good
…                …
28454   strike, good
28455     ball, good
28456   strike, good
28457     ball, good
28458     ball, good

[28459 rows x 6 columns]
```

[44]:
```
#### BRETT GARDNER STATS
import math
print("============================ BRETT GARDNER DATA␣
 ↪============================ \n")
newGardyT = [x for x in gardyT if math.isnan(x) == False] #### removes nan
newGardyB = [y for y in gardyB if math.isnan(y) == False] #### removes nan
avgTop = round(Average(newGardyT),3)
avgBot = round(Average(newGardyB),3)
Gardy = []
S_PitchCount = 0 # pitch count
S_Count = 0 # strike count
B_Count = 0 # ball count
goodStrikes = [] #list of tuples - (xpos, zpos)
badStrikes = []
goodBalls =[]
badBalls = []
for x in range(len(newdf)):  #### add pitch coordinates as tuples to color code␣
 ↪plot
    if ((newdf.iloc[x]).iloc[0]) == "Brett Gardner":
        Gardy.append(newdf.loc[x, :].values.flatten().tolist()) # add row to␣
↪new list
        S_PitchCount += 1
        if ((newdf.iloc[x]).iloc[5]) == "strike, good" or ((newdf.iloc[x]).
↪iloc[5]) == "strike, bad":
            S_Count += 1
            if ((newdf.iloc[x]).iloc[5]) == "strike, good":
                goodStrikes.append(((newdf.iloc[x]).iloc[1], (newdf.iloc[x]).
↪iloc[2]))
            else:
                badStrikes.append(((newdf.iloc[x]).iloc[1], (newdf.iloc[x]).
↪iloc[2]))
        else:
            B_Count += 1
            if ((newdf.iloc[x]).iloc[5]) == "ball, good":
                goodBalls.append(((newdf.iloc[x]).iloc[1], (newdf.iloc[x]).
↪iloc[2]))
            else:
                badBalls.append(((newdf.iloc[x]).iloc[1], (newdf.iloc[x]).
↪iloc[2]))

#### Gardner Data Frame
GardyDF = pd.DataFrame(Gardy, columns = ['Name','xPos', 'zPos', 'Pitch Type',␣
↪'Given Description', 'Classification'])
#print(GardyDF.head(len(GardyDF.index)).to_string())
print("FA - Fastball, CU - Curveball, FT - Two-seam Fastball, CH - Changeup,␣
↪\nFC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,\nFF - Four-seam␣
↪Fastball \n")
print(GardyDF)
print()

percent_Good = len(goodStrikes) / S_Count
percent_Bad = len(goodBalls) /  B_Count
print("% of "good" strikes: ", round(percent_Good,3) )
print("% of "good" balls:",  round(percent_Bad,3) )
print("(% good strikes) + (% good balls): ", round((percent_Good +␣
↪percent_Bad),3) )

fig = plt.figure()
ax1 = fig.add_subplot(111)
ax1.set_xlim(-1,1)
ax1.set_ylim(0,5)
for x in range(len(goodStrikes)):
    p1 = ax1.scatter(goodStrikes[x][0],goodStrikes[x][1], s=10, c='g',␣
↪marker="o")
for y in range(len(badStrikes)):
    p2 = ax1.scatter(badStrikes[y][0], badStrikes[y][1], s=10, c='r',␣
 ↪marker="o")
plt.title("Brett Gardner Strike Data", fontweight ='bold',size=18)
plt.legend([p1,p2], ["good strikes", "bad strikes"])
ax1.add_patch(Rectangle((-0.71, avgBot), 1.42, avgTop-avgBot ,edgecolor =␣
 ↪'red',fill=False,lw=2))
plt.show()

fig = plt.figure()
ax2 = fig.add_subplot(111)
ax2.set_xlim(-2.75,2.75)
ax2.set_ylim(0,5)
for x in range(len(goodBalls)):
    p21 = ax2.scatter(goodBalls[x][0],goodBalls[x][1], s=10, c='g', marker="o")
for y in range(len(badBalls)):
    p22 = ax2.scatter(badBalls[y][0], badBalls[y][1], s=10, c='r', marker="o", )
plt.title("Brett Gardner Ball Data", fontweight ='bold',size=18)
plt.legend([p21,p22], ["good balls", "bad balls"])
ax2.add_patch(Rectangle((-0.71, avgBot), 1.42, avgTop-avgBot ,edgecolor =␣
 ↪'red',fill=False,lw=2))
plt.show()
```

```
============================ BRETT GARDNER DATA ============================

FA - Fastball, CU - Curveball, FT - Two-seam Fastball, CH - Changeup,
FC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,
FF - Four-seam Fastball

               Name  xPos  zPos Pitch Type Given Description Classification
0     Brett Gardner  0.14  1.69         CH   swinging_strike   strike, good
1     Brett Gardner  0.18  2.85         FF              foul   strike, good
2     Brett Gardner -0.60  2.76         FF   swinging_strike   strike, good
3     Brett Gardner  1.50  3.60         FC              ball     ball, good
4     Brett Gardner  0.15  2.71         FF     called_strike    strike, bad
…               …   …    …          …               …           …
2539  Brett Gardner -1.46  2.56         FT              ball     ball, good
2540  Brett Gardner -0.72  1.82         SL     called_strike     ball, good
2541  Brett Gardner -0.03  3.61         FF   swinging_strike      ball, bad
2542  Brett Gardner -0.66  1.63         SL     called_strike    strike, bad
2543  Brett Gardner -0.14  1.72         FF     called_strike    strike, bad

[2544 rows x 6 columns]

% of "good" strikes:  0.57
% of "good" balls: 0.733
(% good strikes) + (% good balls):  1.304
```


Brett Gardner Strike Data
good strikes
bad strikes
5
4
3
2
1
0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


Brett Gardner Ball Data
good balls
bad balls
5
4
3
2
1
0
−2
−1
0
1
2

```
[39]: #### ALL NYY PLAYER STATS
from IPython.core.display import display, HTML
class color:
   PURPLE = '\033[95m'
   CYAN = '\033[96m'
   DARKCYAN = '\033[36m'
   BLUE = '\033[94m'
   GREEN = '\033[92m'
   YELLOW = '\033[93m'
   RED = '\033[91m'
   BOLD = '\033[1m'
   UNDERLINE = '\033[4m'
   END = '\033[0m'

display(HTML("<style>.container { width:100% !important; }</style>"))
print("============================ NYY PLAYER STAT DATA␣
↪============================ \n")
nyyList = []
for yank in playerNums:
    num_pitches = 0
    num_strikes = 0
    num_balls = 0
    num_good_strikes = 0
    num_good_balls = 0
    for x in range(len(newdf)):
        if ((newdf.iloc[x]).iloc[0]) == yank:
            num_pitches += 1
            if ((newdf.iloc[x]).iloc[5]) == "strike, good" or ((newdf.iloc[x]).
↪iloc[5]) == "strike, bad":
                num_strikes += 1
                if ((newdf.iloc[x]).iloc[5]) == "strike, good":
                    num_good_strikes += 1
            else:
                num_balls += 1
                if ((newdf.iloc[x]).iloc[5]) == "ball, good":
                    num_good_balls += 1
    if num_strikes != 0 and num_balls != 0 and num_pitches >= 100:
        p_Good = num_good_strikes / num_strikes
        p_Bad = num_good_balls / num_balls
        nyyList.append([yank,round(num_pitches,3) ,round(num_strikes,3),␣
↪round(num_balls,3), round(p_Good,3),round(p_Bad,3),round((p_Good +␣
↪p_Bad),3)])

NYYDF = pd.DataFrame(nyyList, columns = ['Name','Pitches', 'Strikes', 'Balls',␣
↪'g_strike', 'g_ball', 'DS'])

ax = NYYDF.plot.barh(x='Name', y='g_strike', figsize=(12, 8)) # Bar graph plot
```

```python
plt.title("Good Strikes", fontweight ='bold',size=18)
plt.ylabel('Name', fontweight='bold', size=12)
plt.axvline(x=0.699, color='r', linestyle='--') # Draws average line
ax2 = NYYDF.plot.barh(x='Name', y='g_ball', figsize=(12, 8)) # Bar graph plot
plt.title("Good Balls", fontweight ='bold',size=18)
plt.ylabel('Name', fontweight='bold', size=12)
plt.axvline(x=0.673, color='r', linestyle='--') # Draws average line

print(color.BOLD + "\t\t\t    Ordered by name" + color.END)
print(NYYDF.sort_values(by='Name', ascending=True), "\n")
print(color.BOLD + "\t\t\t Ordered by pitches" + color.END)
sort = NYYDF.sort_values(by='Pitches', ascending=False)
print(sort, "\n")
print(color.BOLD + "\t\t\t Ordered by good strikes" + color.END)
sort1 = NYYDF.sort_values(by='g_strike', ascending=False)
print(sort1, "\n")
print(color.BOLD + "\t\t\t Ordered by good balls" + color.END)
sort2 = NYYDF.sort_values(by='g_ball', ascending=False)
print(sort2, "\n")
print(color.BOLD + "\t\t\t   Ordered by discernment score" + color.END)
sort3 = NYYDF.sort_values(by='DS', ascending=False)
print(sort3, "\n")

g_strike_AVG = NYYDF["g_strike"].mean() #### Gets average of whole column
g_ball_AVG = NYYDF["g_ball"].mean()
print("Team Good Strike Average: ", round(g_strike_AVG,3), "\nTeam Good Ball␣
 ↪Average: ", round(g_ball_AVG,3))
```

```
<IPython.core.display.HTML object>


============================ NYY PLAYER STAT DATA ============================


                      Ordered by name
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
11        Aaron Hicks     1151      399    752     0.657   0.726  1.383
7         Aaron Judge     2073      727   1346     0.693   0.724  1.417
13      Austin Romine      871      356    515     0.806   0.598  1.404
6       Brett Gardner     2544      991   1553     0.570   0.733  1.304
22     Breyvic Valera      205       78    127     0.744   0.724  1.468
10     Cameron Maybin     1154      445    709     0.649   0.722  1.372
12      Clint Frazier      976      382    594     0.725   0.742  1.468
2         DJ LeMahieu     2618     1021   1597     0.670   0.681  1.351
3      Didi Gregorius     1504      542    962     0.782   0.570  1.352
8   Edwin Encarnacion     2262      826   1436     0.688   0.698  1.385
0        Gary Sanchez     2053      647   1406     0.604   0.632  1.237
17  Giancarlo Stanton      395      121    274     0.612   0.686  1.298
4         Gio Urshela     1908      727   1181     0.761   0.556  1.317
```

```
9      Gleyber Torres     2551      953   1598      0.781    0.634  1.415
21          Greg Bird      206       76    130      0.632    0.677  1.309
16    Kendrys Morales      833      313    520      0.633    0.748  1.381
19    Kyle Higashioka      246      103    143      0.845    0.552  1.397
1           Luke Voit     2016      731   1285      0.804    0.715  1.520
20     Miguel Andujar      175       61    114      0.689    0.544  1.232
14          Mike Ford      657      245    412      0.649    0.748  1.397
5       Mike Tauchman     1288      491    797      0.731    0.755  1.486
18     Thairo Estrada      227       95    132      0.632    0.614  1.245
15         Tyler Wade      388      153    235      0.725    0.694  1.419
```

Ordered by pitches

```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
2         DJ LeMahieu     2618     1021   1597     0.670   0.681  1.351
9      Gleyber Torres     2551      953   1598     0.781   0.634  1.415
6       Brett Gardner     2544      991   1553     0.570   0.733  1.304
8   Edwin Encarnacion     2262      826   1436     0.688   0.698  1.385
7         Aaron Judge     2073      727   1346     0.693   0.724  1.417
0        Gary Sanchez     2053      647   1406     0.604   0.632  1.237
1           Luke Voit     2016      731   1285     0.804   0.715  1.520
4         Gio Urshela     1908      727   1181     0.761   0.556  1.317
3      Didi Gregorius     1504      542    962     0.782   0.570  1.352
5       Mike Tauchman     1288      491    797     0.731   0.755  1.486
10     Cameron Maybin     1154      445    709     0.649   0.722  1.372
11        Aaron Hicks     1151      399    752     0.657   0.726  1.383
12      Clint Frazier      976      382    594     0.725   0.742  1.468
13      Austin Romine      871      356    515     0.806   0.598  1.404
16    Kendrys Morales      833      313    520     0.633   0.748  1.381
14          Mike Ford      657      245    412     0.649   0.748  1.397
17  Giancarlo Stanton      395      121    274     0.612   0.686  1.298
15         Tyler Wade      388      153    235     0.725   0.694  1.419
19    Kyle Higashioka      246      103    143     0.845   0.552  1.397
18     Thairo Estrada      227       95    132     0.632   0.614  1.245
21          Greg Bird      206       76    130     0.632   0.677  1.309
22     Breyvic Valera      205       78    127     0.744   0.724  1.468
20     Miguel Andujar      175       61    114     0.689   0.544  1.232
```

Ordered by good strikes

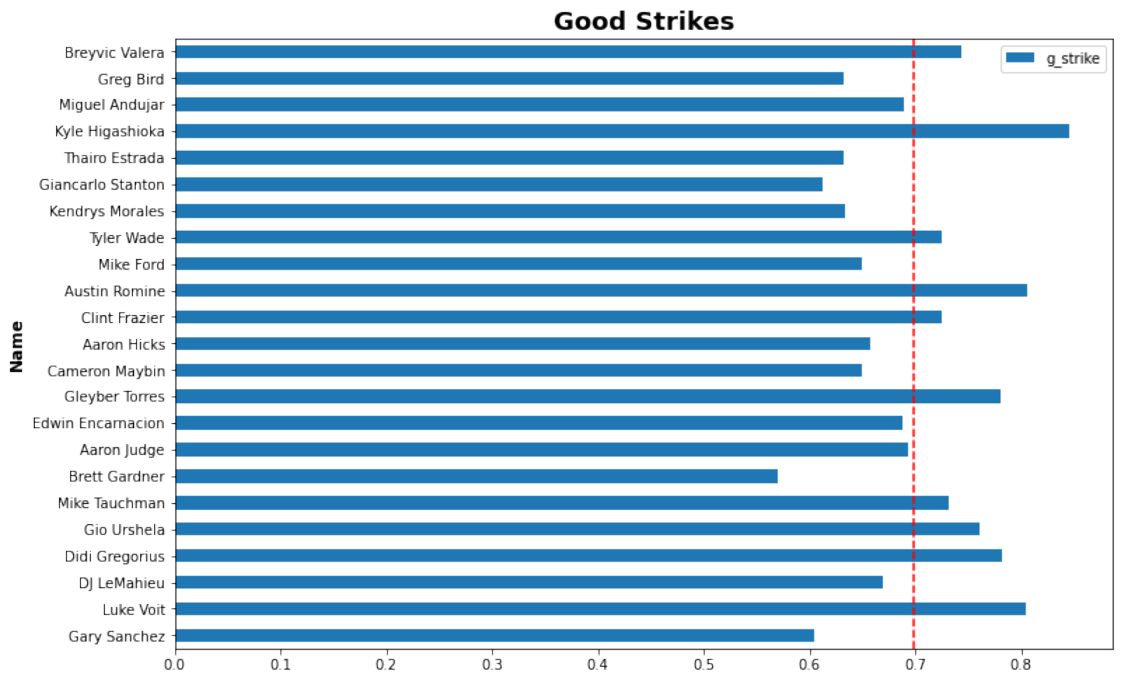
```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
19    Kyle Higashioka      246      103    143     0.845   0.552  1.397
13      Austin Romine      871      356    515     0.806   0.598  1.404
1           Luke Voit     2016      731   1285     0.804   0.715  1.520
3      Didi Gregorius     1504      542    962     0.782   0.570  1.352
9      Gleyber Torres     2551      953   1598     0.781   0.634  1.415
4         Gio Urshela     1908      727   1181     0.761   0.556  1.317
22     Breyvic Valera      205       78    127     0.744   0.724  1.468
5       Mike Tauchman     1288      491    797     0.731   0.755  1.486
15         Tyler Wade      388      153    235     0.725   0.694  1.419
```

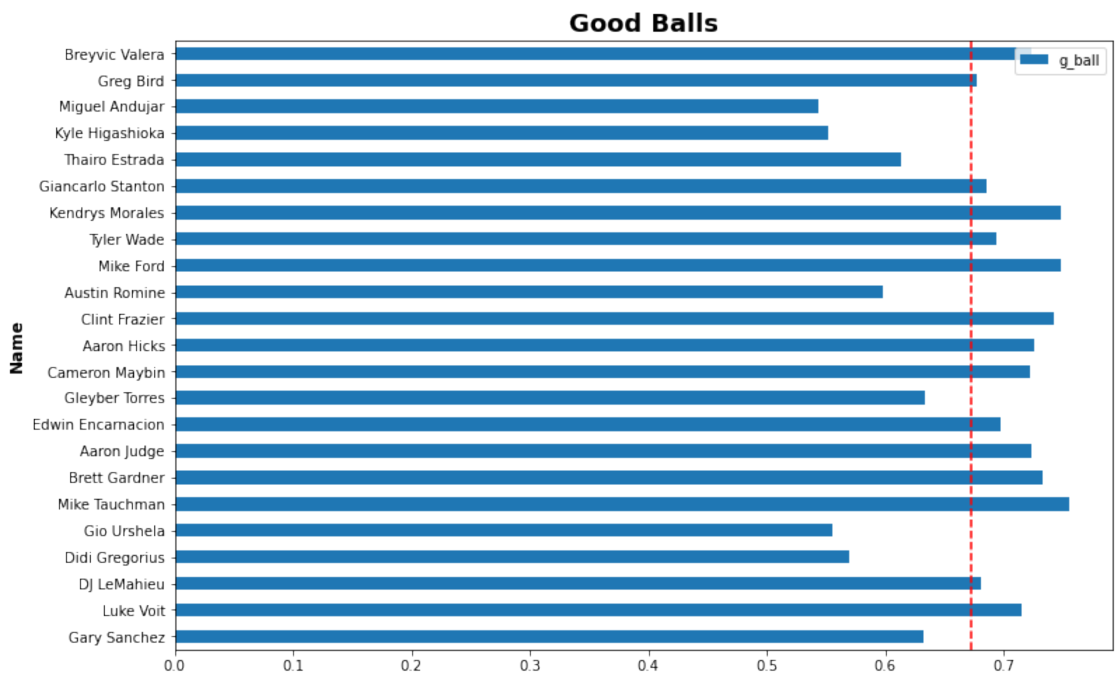
```
12       Clint Frazier      976      382    594     0.725   0.742  1.468
7          Aaron Judge     2073      727   1346     0.693   0.724  1.417
20      Miguel Andujar      175       61    114     0.689   0.544  1.232
8   Edwin Encarnacion     2262      826   1436     0.688   0.698  1.385
2          DJ LeMahieu     2618     1021   1597     0.670   0.681  1.351
11         Aaron Hicks     1151      399    752     0.657   0.726  1.383
14           Mike Ford      657      245    412     0.649   0.748  1.397
10      Cameron Maybin     1154      445    709     0.649   0.722  1.372
16     Kendrys Morales      833      313    520     0.633   0.748  1.381
18      Thairo Estrada      227       95    132     0.632   0.614  1.245
21           Greg Bird      206       76    130     0.632   0.677  1.309
17  Giancarlo Stanton      395      121    274     0.612   0.686  1.298
0          Gary Sanchez    2053      647   1406     0.604   0.632  1.237
6        Brett Gardner     2544      991   1553     0.570   0.733  1.304
```

```
                  Ordered by good balls
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
5       Mike Tauchman     1288      491    797     0.731   0.755  1.486
16     Kendrys Morales      833      313    520     0.633   0.748  1.381
14           Mike Ford      657      245    412     0.649   0.748  1.397
12       Clint Frazier      976      382    594     0.725   0.742  1.468
6        Brett Gardner     2544      991   1553     0.570   0.733  1.304
11         Aaron Hicks     1151      399    752     0.657   0.726  1.383
7          Aaron Judge     2073      727   1346     0.693   0.724  1.417
22      Breyvic Valera      205       78    127     0.744   0.724  1.468
10      Cameron Maybin     1154      445    709     0.649   0.722  1.372
1            Luke Voit     2016      731   1285     0.804   0.715  1.520
8   Edwin Encarnacion     2262      826   1436     0.688   0.698  1.385
15          Tyler Wade      388      153    235     0.725   0.694  1.419
17  Giancarlo Stanton      395      121    274     0.612   0.686  1.298
2          DJ LeMahieu     2618     1021   1597     0.670   0.681  1.351
21           Greg Bird      206       76    130     0.632   0.677  1.309
9       Gleyber Torres     2551      953   1598     0.781   0.634  1.415
0         Gary Sanchez     2053      647   1406     0.604   0.632  1.237
18      Thairo Estrada      227       95    132     0.632   0.614  1.245
13      Austin Romine      871      356    515     0.806   0.598  1.404
3      Didi Gregorius     1504      542    962     0.782   0.570  1.352
4         Gio Urshela     1908      727   1181     0.761   0.556  1.317
19    Kyle Higashioka      246      103    143     0.845   0.552  1.397
20      Miguel Andujar      175       61    114     0.689   0.544  1.232
```

```
                    Ordered by discernment score
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
1            Luke Voit     2016      731   1285     0.804   0.715  1.520
5       Mike Tauchman     1288      491    797     0.731   0.755  1.486
22      Breyvic Valera      205       78    127     0.744   0.724  1.468
12       Clint Frazier      976      382    594     0.725   0.742  1.468
15          Tyler Wade      388      153    235     0.725   0.694  1.419
```

```
7          Aaron Judge     2073       727    1346      0.693    0.724   1.417
9       Gleyber Torres     2551       953    1598      0.781    0.634   1.415
13       Austin Romine      871       356     515      0.806    0.598   1.404
19     Kyle Higashioka      246       103     143      0.845    0.552   1.397
14           Mike Ford      657       245     412      0.649    0.748   1.397
8    Edwin Encarnacion     2262       826    1436      0.688    0.698   1.385
11         Aaron Hicks     1151       399     752      0.657    0.726   1.383
16     Kendrys Morales      833       313     520      0.633    0.748   1.381
10      Cameron Maybin     1154       445     709      0.649    0.722   1.372
3       Didi Gregorius     1504       542     962      0.782    0.570   1.352
2          DJ LeMahieu     2618      1021    1597      0.670    0.681   1.351
4          Gio Urshela     1908       727    1181      0.761    0.556   1.317
21           Greg Bird      206        76     130      0.632    0.677   1.309
6        Brett Gardner     2544       991    1553      0.570    0.733   1.304
17   Giancarlo Stanton      395       121     274      0.612    0.686   1.298
18      Thairo Estrada      227        95     132      0.632    0.614   1.245
0         Gary Sanchez     2053       647    1406      0.604    0.632   1.237
20      Miguel Andujar      175        61     114      0.689    0.544   1.232

Team Good Strike Average:  0.699
Team Good Ball Average:  0.673
```





[44]:
```
#### ONLY 3-2 COUNT PITCHES
print("============================ NYY 3-2 COUNT DATA␣
↪============================ \n")
YankAtBats32 = []
for z in range(len(YankAtBats)):
    if Count_3_2[z] == 1:
        YankAtBats32.append(YankAtBats[z])

YankAtBats32df = pd.DataFrame(YankAtBats32[0:len(YankAtBats32)-5], columns =␣
↪['Name', 'xPos', 'zPos', 'Pitch Type', 'Given Description',␣
↪'Classification'])
#print(YankAtBats32df,"\n")

nyyList2 = []
for yank in playerNums:
    num_pitches = 0
    num_strikes = 0
    num_balls = 0
    num_good_strikes = 0
    num_good_balls = 0
    for x in range(len(YankAtBats32df)):
        if ((YankAtBats32df.iloc[x]).iloc[0]) == yank:
            num_pitches += 1
            if ((YankAtBats32df.iloc[x]).iloc[5]) == "strike, good" or␣
↪((YankAtBats32df.iloc[x]).iloc[5]) == "strike, bad":
```

```
                num_strikes += 1
                if ((YankAtBats32df.iloc[x]).iloc[5]) == "strike, good":
                    num_good_strikes += 1
            else:
                num_balls += 1
                if ((YankAtBats32df.iloc[x]).iloc[5]) == "ball, good":
                    num_good_balls += 1
    if num_strikes != 0 and num_balls != 0 and num_pitches >= 50:  #### min 50␣
↪pitches
        p_Good = num_good_strikes / num_strikes
        p_Bad = num_good_balls / num_balls
        nyyList2.append([yank,round(num_pitches,3) ,round(num_strikes,3),␣
↪round(num_balls,3), round(p_Good,3),round(p_Bad,3),round((p_Good +␣
↪p_Bad),3)])

YankAtBats32df = pd.DataFrame(nyyList2, columns = ['Name','Pitches', 'Strikes',␣
↪'Balls', 'g_strike', 'g_ball', 'DS'])
print(color.BOLD + "\t\t\t    Ordered by name" + color.END)
print(YankAtBats32df.sort_values(by='Name', ascending=True), "\n")
print(color.BOLD + "\t\t\t Ordered by pitches" + color.END)
sort = YankAtBats32df.sort_values(by='Pitches', ascending=False)
print(sort, "\n")
print(color.BOLD + "\t\t\t Ordered by good strikes" + color.END)
sort1 = YankAtBats32df.sort_values(by='g_strike', ascending=False)
print(sort1, "\n")
print(color.BOLD + "\t\t\t Ordered by good balls" + color.END)
sort2 = YankAtBats32df.sort_values(by='g_ball', ascending=False)
print(sort2, "\n")
print(color.BOLD + "\t\t\t   Ordered by discernment score" + color.END)
sort3 = YankAtBats32df.sort_values(by='DS', ascending=False)
print(sort3, "\n")
g_strike_AVG = YankAtBats32df["g_strike"].mean() #### Gets average of whole␣
↪column
g_ball_AVG = YankAtBats32df["g_ball"].mean()
print("Team Good Strike Average: ", round(g_strike_AVG,3), "\nTeam Good Ball␣
↪Average: ", round(g_ball_AVG,3))
```

```
============================ NYY 3-2 COUNT DATA ============================

                       Ordered by name
                Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
11       Aaron Hicks       72       27     45     0.852   0.533  1.385
7        Aaron Judge      130       58     72     0.879   0.625  1.504
6      Brett Gardner      137       86     51     0.953   0.333  1.287
10    Cameron Maybin       78       41     37     0.902   0.432  1.335
12     Clint Frazier       51       22     29     0.955   0.241  1.196
2        DJ LeMahieu      125       51     74     0.922   0.446  1.368
```

```
3      Didi Gregorius       69        34     35     1.000   0.314  1.314
8   Edwin Encarnacion      163        71     92     0.944   0.467  1.411
0        Gary Sanchez      126        53     73     0.943   0.288  1.231
4         Gio Urshela       65        28     37     0.964   0.432  1.397
9      Gleyber Torres      146        64     82     0.969   0.366  1.335
13    Kendrys Morales       58        28     30     0.929   0.500  1.429
1           Luke Voit      109        37     72     0.757   0.653  1.410
5       Mike Tauchman       81        33     48     0.939   0.521  1.460

                  Ordered by pitches
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
8   Edwin Encarnacion      163       71     92     0.944   0.467  1.411
9      Gleyber Torres      146       64     82     0.969   0.366  1.335
6       Brett Gardner      137       86     51     0.953   0.333  1.287
7         Aaron Judge      130       58     72     0.879   0.625  1.504
0        Gary Sanchez      126       53     73     0.943   0.288  1.231
2         DJ LeMahieu      125       51     74     0.922   0.446  1.368
1           Luke Voit      109       37     72     0.757   0.653  1.410
5       Mike Tauchman       81       33     48     0.939   0.521  1.460
10     Cameron Maybin       78       41     37     0.902   0.432  1.335
11        Aaron Hicks       72       27     45     0.852   0.533  1.385
3      Didi Gregorius       69       34     35     1.000   0.314  1.314
4         Gio Urshela       65       28     37     0.964   0.432  1.397
13    Kendrys Morales       58       28     30     0.929   0.500  1.429
12      Clint Frazier       51       22     29     0.955   0.241  1.196

                  Ordered by good strikes
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
3      Didi Gregorius       69       34     35     1.000   0.314  1.314
9      Gleyber Torres      146       64     82     0.969   0.366  1.335
4         Gio Urshela       65       28     37     0.964   0.432  1.397
12      Clint Frazier       51       22     29     0.955   0.241  1.196
6       Brett Gardner      137       86     51     0.953   0.333  1.287
8   Edwin Encarnacion      163       71     92     0.944   0.467  1.411
0        Gary Sanchez      126       53     73     0.943   0.288  1.231
5       Mike Tauchman       81       33     48     0.939   0.521  1.460
13    Kendrys Morales       58       28     30     0.929   0.500  1.429
2         DJ LeMahieu      125       51     74     0.922   0.446  1.368
10     Cameron Maybin       78       41     37     0.902   0.432  1.335
7         Aaron Judge      130       58     72     0.879   0.625  1.504
11        Aaron Hicks       72       27     45     0.852   0.533  1.385
1           Luke Voit      109       37     72     0.757   0.653  1.410

                  Ordered by good balls
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
1           Luke Voit      109       37     72     0.757   0.653  1.410
7         Aaron Judge      130       58     72     0.879   0.625  1.504
11        Aaron Hicks       72       27     45     0.852   0.533  1.385
```

```
5         Mike Tauchman       81       33     48      0.939    0.521   1.460
13     Kendrys Morales        58       28     30      0.929    0.500   1.429
8    Edwin Encarnacion       163       71     92      0.944    0.467   1.411
2             DJ LeMahieu    125       51     74      0.922    0.446   1.368
4             Gio Urshela     65       28     37      0.964    0.432   1.397
10       Cameron Maybin       78       41     37      0.902    0.432   1.335
9        Gleyber Torres      146       64     82      0.969    0.366   1.335
6         Brett Gardner      137       86     51      0.953    0.333   1.287
3        Didi Gregorius       69       34     35      1.000    0.314   1.314
0           Gary Sanchez     126       53     73      0.943    0.288   1.231
12        Clint Frazier       51       22     29      0.955    0.241   1.196

                      Ordered by discernment score
                   Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
7           Aaron Judge      130       58     72     0.879   0.625  1.504
5         Mike Tauchman       81       33     48     0.939   0.521  1.460
13     Kendrys Morales        58       28     30     0.929   0.500  1.429
8    Edwin Encarnacion       163       71     92     0.944   0.467  1.411
1             Luke Voit      109       37     72     0.757   0.653  1.410
4           Gio Urshela       65       28     37     0.964   0.432  1.397
11          Aaron Hicks       72       27     45     0.852   0.533  1.385
2           DJ LeMahieu      125       51     74     0.922   0.446  1.368
9        Gleyber Torres      146       64     82     0.969   0.366  1.335
10       Cameron Maybin       78       41     37     0.902   0.432  1.335
3        Didi Gregorius       69       34     35     1.000   0.314  1.314
6         Brett Gardner      137       86     51     0.953   0.333  1.287
0          Gary Sanchez      126       53     73     0.943   0.288  1.231
12        Clint Frazier       51       22     29     0.955   0.241  1.196

Team Good Strike Average:  0.922
Team Good Ball Average:  0.439
```

[45]:
```python
#### ONLY FAST BALLS (2 and 4 seam)
print("============================ NYY FASTBALL DATA␣
↪============================ \n")
nyyList3 = []
for yank in playerNums:
    num_pitches = 0
    num_strikes = 0
    num_balls = 0
    num_good_strikes = 0
    num_good_balls = 0
    for x in range(len(newdf)):
        if ((newdf.iloc[x]).iloc[0]) == yank and (((newdf.iloc[x]).iloc[3]) ==␣
↪"FT" or ((newdf.iloc[x]).iloc[3]) == "FF"):
            num_pitches += 1
            if ((newdf.iloc[x]).iloc[5]) == "strike, good" or ((newdf.iloc[x]).
↪iloc[5]) == "strike, bad":
                num_strikes += 1
                if ((newdf.iloc[x]).iloc[5]) == "strike, good":
                    num_good_strikes += 1
            else:
                num_balls += 1
                if ((newdf.iloc[x]).iloc[5]) == "ball, good":
                    num_good_balls += 1
    if num_strikes != 0 and num_balls != 0 and num_pitches >= 100:  #### Only␣
↪using 100 here
        p_Good = num_good_strikes / num_strikes
        p_Bad = num_good_balls / num_balls
        nyyList3.append([yank,round(num_pitches,3) ,round(num_strikes,3),␣
↪round(num_balls,3), round(p_Good,3),round(p_Bad,3),round((p_Good +␣
↪p_Bad),3)])

FastBallsDF = pd.DataFrame(nyyList3, columns = ['Name','Pitches', 'Strikes',␣
↪'Balls', 'g_strike', 'g_ball', 'DS'])

print(color.BOLD + "\t\t\t    Ordered by name" + color.END)
print(FastBallsDF.sort_values(by='Name', ascending=True), "\n")
print(color.BOLD + "\t\t\t Ordered by pitches" + color.END)
sort = FastBallsDF.sort_values(by='Pitches', ascending=False)
print(sort, "\n")
print(color.BOLD + "\t\t\t Ordered by good strikes" + color.END)
sort1 = FastBallsDF.sort_values(by='g_strike', ascending=False)
print(sort1, "\n")
print(color.BOLD + "\t\t\t Ordered by good balls" + color.END)
sort2 = FastBallsDF.sort_values(by='g_ball', ascending=False)
print(sort2, "\n")
print(color.BOLD + "\t\t\t   Ordered by discernment score" + color.END)
sort3 = FastBallsDF.sort_values(by='DS', ascending=False)
print(sort3, "\n")
g_strike_AVG = FastBallsDF["g_strike"].mean() #### Gets average of whole column
g_ball_AVG = FastBallsDF["g_ball"].mean()
print("Team Good Strike Average: ", round(g_strike_AVG,3), "\nTeam Good Ball␣
↪Average: ", round(g_ball_AVG,3))
```

```
============================ NYY FASTBALL DATA ============================

                        Ordered by name
                  Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
11         Aaron Hicks      471      188    283     0.654   0.749  1.403
7          Aaron Judge      800      341    459     0.707   0.734  1.441
13       Austin Romine      352      151    201     0.834   0.672  1.506
6        Brett Gardner     1218      518    700     0.573   0.754  1.328
```

```
10     Cameron Maybin      459      201     258     0.726   0.752  1.478
12      Clint Frazier      424      178     246     0.669   0.793  1.461
2         DJ LeMahieu     1278      563     715     0.631   0.737  1.368
3      Didi Gregorius      628      256     372     0.734   0.548  1.283
8   Edwin Encarnacion      837      344     493     0.680   0.680  1.360
0        Gary Sanchez      788      301     487     0.671   0.639  1.310
17  Giancarlo Stanton      179       68     111     0.588   0.703  1.291
4         Gio Urshela      771      340     431     0.776   0.573  1.350
9      Gleyber Torres     1014      428     586     0.820   0.662  1.482
19           Greg Bird      102       42      60     0.524   0.700  1.224
16    Kendrys Morales      334      145     189     0.607   0.767  1.374
18    Kyle Higashioka      100       47      53     0.830   0.509  1.339
1           Luke Voit      810      336     474     0.821   0.762  1.583
14          Mike Ford      242      101     141     0.723   0.702  1.425
5       Mike Tauchman      612      292     320     0.740   0.825  1.565
15         Tyler Wade      166       68      98     0.721   0.786  1.506
```

Ordered by pitches

```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
2         DJ LeMahieu     1278      563    715     0.631   0.737  1.368
6       Brett Gardner     1218      518    700     0.573   0.754  1.328
9      Gleyber Torres     1014      428    586     0.820   0.662  1.482
8   Edwin Encarnacion      837      344    493     0.680   0.680  1.360
1           Luke Voit      810      336    474     0.821   0.762  1.583
7         Aaron Judge      800      341    459     0.707   0.734  1.441
0        Gary Sanchez      788      301    487     0.671   0.639  1.310
4         Gio Urshela      771      340    431     0.776   0.573  1.350
3      Didi Gregorius      628      256    372     0.734   0.548  1.283
5       Mike Tauchman      612      292    320     0.740   0.825  1.565
11        Aaron Hicks      471      188    283     0.654   0.749  1.403
10     Cameron Maybin      459      201    258     0.726   0.752  1.478
12      Clint Frazier      424      178    246     0.669   0.793  1.461
13      Austin Romine      352      151    201     0.834   0.672  1.506
16    Kendrys Morales      334      145    189     0.607   0.767  1.374
14          Mike Ford      242      101    141     0.723   0.702  1.425
17  Giancarlo Stanton      179       68    111     0.588   0.703  1.291
15         Tyler Wade      166       68     98     0.721   0.786  1.506
19          Greg Bird      102       42     60     0.524   0.700  1.224
18    Kyle Higashioka      100       47     53     0.830   0.509  1.339
```

Ordered by good strikes

```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
13      Austin Romine      352      151    201     0.834   0.672  1.506
18    Kyle Higashioka      100       47     53     0.830   0.509  1.339
1           Luke Voit      810      336    474     0.821   0.762  1.583
9      Gleyber Torres     1014      428    586     0.820   0.662  1.482
4         Gio Urshela      771      340    431     0.776   0.573  1.350
5       Mike Tauchman      612      292    320     0.740   0.825  1.565
```

```
3      Didi Gregorius      628       256    372      0.734    0.548  1.283
10     Cameron Maybin      459       201    258      0.726    0.752  1.478
14          Mike Ford      242       101    141      0.723    0.702  1.425
15         Tyler Wade      166        68     98      0.721    0.786  1.506
7         Aaron Judge      800       341    459      0.707    0.734  1.441
8   Edwin Encarnacion      837       344    493      0.680    0.680  1.360
0        Gary Sanchez      788       301    487      0.671    0.639  1.310
12      Clint Frazier      424       178    246      0.669    0.793  1.461
11        Aaron Hicks      471       188    283      0.654    0.749  1.403
2         DJ LeMahieu     1278       563    715      0.631    0.737  1.368
16    Kendrys Morales      334       145    189      0.607    0.767  1.374
17  Giancarlo Stanton      179        68    111      0.588    0.703  1.291
6       Brett Gardner     1218       518    700      0.573    0.754  1.328
19          Greg Bird      102        42     60      0.524    0.700  1.224
```

Ordered by good balls

```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
5       Mike Tauchman      612       292    320      0.740    0.825  1.565
12      Clint Frazier      424       178    246      0.669    0.793  1.461
15         Tyler Wade      166        68     98      0.721    0.786  1.506
16    Kendrys Morales      334       145    189      0.607    0.767  1.374
1           Luke Voit      810       336    474      0.821    0.762  1.583
6       Brett Gardner     1218       518    700      0.573    0.754  1.328
10     Cameron Maybin      459       201    258      0.726    0.752  1.478
11        Aaron Hicks      471       188    283      0.654    0.749  1.403
2         DJ LeMahieu     1278       563    715      0.631    0.737  1.368
7         Aaron Judge      800       341    459      0.707    0.734  1.441
17  Giancarlo Stanton      179        68    111      0.588    0.703  1.291
14          Mike Ford      242       101    141      0.723    0.702  1.425
19          Greg Bird      102        42     60      0.524    0.700  1.224
8   Edwin Encarnacion      837       344    493      0.680    0.680  1.360
13      Austin Romine      352       151    201      0.834    0.672  1.506
9      Gleyber Torres     1014       428    586      0.820    0.662  1.482
0        Gary Sanchez      788       301    487      0.671    0.639  1.310
4         Gio Urshela      771       340    431      0.776    0.573  1.350
3      Didi Gregorius      628       256    372      0.734    0.548  1.283
18    Kyle Higashioka      100        47     53      0.830    0.509  1.339
```

Ordered by discernment score

```
                 Name  Pitches  Strikes  Balls  g_strike  g_ball     DS
1           Luke Voit      810       336    474      0.821    0.762  1.583
5       Mike Tauchman      612       292    320      0.740    0.825  1.565
13      Austin Romine      352       151    201      0.834    0.672  1.506
15         Tyler Wade      166        68     98      0.721    0.786  1.506
9      Gleyber Torres     1014       428    586      0.820    0.662  1.482
10     Cameron Maybin      459       201    258      0.726    0.752  1.478
12      Clint Frazier      424       178    246      0.669    0.793  1.461
7         Aaron Judge      800       341    459      0.707    0.734  1.441
```

```
14          Mike Ford      242      101     141      0.723    0.702  1.425
11        Aaron Hicks      471      188     283      0.654    0.749  1.403
16    Kendrys Morales      334      145     189      0.607    0.767  1.374
2         DJ LeMahieu     1278      563     715      0.631    0.737  1.368
8   Edwin Encarnacion      837      344     493      0.680    0.680  1.360
4         Gio Urshela      771      340     431      0.776    0.573  1.350
18    Kyle Higashioka      100       47      53      0.830    0.509  1.339
6       Brett Gardner     1218      518     700      0.573    0.754  1.328
0        Gary Sanchez      788      301     487      0.671    0.639  1.310
17  Giancarlo Stanton      179       68     111      0.588    0.703  1.291
3      Didi Gregorius      628      256     372      0.734    0.548  1.283
19          Greg Bird      102       42      60      0.524    0.700  1.224

Team Good Strike Average:  0.701
Team Good Ball Average:  0.702
```

[43]:
```python
#CU - Curveball, FT - Two-seam Fastball, CH - Changeup,
#FC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,
#FF - Four-seam Fastball
FB = 0
CU = 0
CH = 0
FC = 0
SL = 0
FS = 0
KN = 0
for x in range(len(newdf)):
        if ((newdf.iloc[x]).iloc[3]) == "FT" or ((newdf.iloc[x]).iloc[3]) ==␣
 ↪"FF":
            FB += 1
        if ((newdf.iloc[x]).iloc[3]) == "CU":
            CU += 1
        if ((newdf.iloc[x]).iloc[3]) == "CH":
            CH += 1
        if ((newdf.iloc[x]).iloc[3]) == "FC":
            FC += 1
        if ((newdf.iloc[x]).iloc[3]) == "SL":
            SL += 1
        if ((newdf.iloc[x]).iloc[3]) == "FS":
            FS += 1
        if ((newdf.iloc[x]).iloc[3]) == "KN":
            KN += 1
print("FB: ", FB)
print("CU: ", CU)
print("CH: ", CH)
print("FC: ", FC)
print("SL: ", SL)
print("FS: ", FS)
print("KN: ", KN)
```

```
FB:  11909
CU:  2559
CH:  2915
FC:  1765
SL:  5583
FS:  581
KN:  0
```

```
[38]: #### STATS FOR ALL OF MLB - Initial Data Frame
from pybaseball import playerid_reverse_lookup
allPlayers = [488726, 514888, 543807, 665742, 543685, 594809, 607208, 645302,␣
 ↪543228, 475582, 545350, 455139, 621043, 452678,
               435062, 670541, 493329, 608324, 502210, 663656, 455117, 543037,␣
 ↪467827, 605452, 572821, 664353, 571578, 571431,
               434671, 435559, 425844, 572191, 649557, 650402, 592450, 518934,␣
 ↪458731, 570482, 596142, 544369, 429665, 543305,
               519317, 669242, 446308, 543939, 572761, 657557, 425877, 542303,␣
 ↪502671, 500874, 622168, 664056, 451594, 668227,
               544931, 656427, 457727, 425794, 453286, 571945, 596847, 502054,␣
 ↪640457, 642715, 595281, 621563, 541645, 519299,
               518595, 621020, 452095, 594807, 455976, 518626, 518692, 645277,␣
 ↪660670, 459964, 435263, 542364, 458708, 457759,
               571970, 572041, 621035, 669257, 571771, 608369, 501896, 641355,␣
 ↪592626, 607461, 621111, 664040, 588751, 465041,
               641712, 572971, 666158, 448179, 443558, 593871, 596146, 543068,␣
 ↪503556, 593934, 650333, 641598, 592696, 595909,
               622110, 650490, 647336, 431145, 628711, 547943, 570731, 592314,␣
 ↪477132, 571740, 593372, 543760, 572033, 543257,
               501981, 595777, 592192, 621566, 656305, 657656, 670712, 664913,␣
 ↪669221, 592325, 606115, 456715, 519346, 460075,
               669374, 519058, 518735, 663757, 519141, 543768, 605540, 641513,␣
 ↪621514, 500135, 543308, 650391, 660162, 572365,
               600869, 474568, 641525, 664901, 570560, 641553, 650489, 606988,␣
 ↪594953, 456078, 641470, 547989, 622682, 408234,
               641313, 595284, 596748, 656514, 656555, 547180, 656371, 623912,␣
 ↪435522, 434158, 543543, 514917, 517369, 446481,
               445988, 542932, 621446, 592407, 621573, 594838, 489149, 645261,␣
 ↪608384, 516770, 643275, 572287, 546990, 624415,
               605233, 571912, 641505, 663993, 621002, 628338, 542583, 620446,␣
 ↪591971, 475253, 642133, 606192, 545341, 640449,
               645801, 425783, 624431, 592261, 642180, 608597, 669256, 596059,␣
 ↪542454, 641432, 462101, 643393, 623520, 518614,
             663538, 605170, 575929, 596825, 664023, 450314, 656803, 608365,␣
↪502706, 656941, 543105, 605244, 573262, 643289,
             576397, 456781, 607680, 664041, 474832, 605131, 542436, 502117,␣
↪621458, 518516, 596103, 446334, 457763, 572073,
             543063, 641914, 622268, 623323, 596129, 553882, 641786, 572039,␣
↪572122, 543592, 641487, 643418, 621450, 620439,
             543829, 605480, 607732, 518653, 622569, 591741, 606299, 518568,␣
↪657434, 592200, 571657, 466320, 624428, 621028,
             596012, 642165, 606157, 553993, 607468, 570481, 571466, 592866,␣
↪641816, 458015, 642086, 425784, 467092, 621512,
             641645, 642708, 434658, 605412, 624424, 624413, 663586, 476704,␣
↪664059, 605204, 607043, 429664, 592789, 453943,
             606992, 640458, 641343, 668663, 664057, 506703, 594694, 644374,␣
↪502273, 623205, 600466, 656185, 605548, 596019,
             608700, 614177, 467793, 605182, 621433, 664926, 600858, 625510,␣
↪596144, 643436, 656811, 641531, 534606, 553902,
             664774, 624585, 641878, 591720, 488771, 460086, 593160, 435622,␣
↪602074, 656541, 621438, 641658, 624513, 460077,
             605486, 592122, 642162, 658069, 606132, 541650, 444489, 641924,␣
↪641857, 475174, 547172, 502517, 571679, 453568,
             656546, 605288, 646240, 605141, 621006, 642851, 592859, 598265,␣
↪593523, 571788, 543877, 444432, 542340, 600524,
             641820, 669720, 593643, 519048, 502110, 593428, 491676, 641796,␣
↪668942, 572233, 500871, 649966, 647304, 642336,
             605113, 527038, 605196, 543376, 543333, 592518, 571976, 594824,␣
↪502481, 605361, 608671, 571875, 592669, 622065,
             620443, 592230, 571718, 664058, 600303, 664702, 596117, 571506,␣
↪642136, 621493, 592743, 594777, 608686, 647351,
             621532, 405395, 605612, 544725, 641477, 608475, 493596, 543510,␣
↪668670, 601713, 595751, 500743, 620453, 621107,
             592761, 605421, 643376, 592273, 641856, 656713, 572228, 642731,␣
↪595981, 543309, 643396, 519222, 518960, 546991,
             592660, 641933, 430935, 519390, 657277, 621005, 664238, 608596,␣
↪640461, 570267, 543377, 547179, 595798, 578428,
             640447, 656725, 643230, 621086, 571467, 608331, 571927, 475247,␣
↪656669, 670032, 596115, 592346, 461829, 448801,
             547004, 506702, 621466, 607752, 572008, 488671, 664034, 642736,␣
↪622534, 623180, 595978, 592662, 572070, 430945,
             665120, 607345, 543045, 624133, 623168, 641154, 664068, 592826,␣
↪624512, 665489, 621219, 595222, 622046, 543101,
             501571, 573186, 547379, 624577, 608723, 668676, 650619, 571448,␣
↪622608, 605200, 518542, 642082, 596105, 608348,
             542882, 641778, 664199, 641684, 608070, 643217, 622786, 607536,␣
↪595881, 664192, 571917, 605397, 500779, 554430,
            657141, 516416, 450306, 622491, 502624, 642221, 605474, 543148,␣
↪502188, 543532, 664119, 659275, 548389, 518792,
            592206, 452657, 642607, 519203, 668804, 643446, 594798, 592767,␣
↪605490, 543475, 622492, 624641, 668800, 518876,
            594577, 501659, 663567, 607572, 664062, 671790, 592567, 543294,␣
↪656605, 605400, 457803, 453284, 571974, 608718,
            518618, 543243, 545121, 641541, 502190, 623993, 600474, 657061,␣
↪642423, 609275, 572362, 608577, 592663, 545333,
            506433, 592178, 553869, 666971, 456501, 615698, 595879, 667498,␣
↪608379, 579328, 666182, 527054, 543548, 643290,
            669160, 607625, 663978, 650859, 594835, 628317, 592863, 606466,␣
↪642547, 492802, 608371, 519144, 622694, 543401,
            543022, 668678, 593958, 518586, 520471, 621529, 643265, 605137,␣
↪501303, 607391, 572816, 622666, 660271, 592885,
            425772, 607333, 457708, 543351, 543302, 543557, 516782, 605446,␣
↪542979, 606956, 643565, 523253, 518633, 605347,
            545361, 641584, 650893, 669222, 664854, 669060, 643615, 641829,␣
↪605280, 545358, 433587, 621097, 656954, 608337,
            593576, 594965, 467055, 607054, 592612, 622772, 612672, 641149,␣
↪467100, 571980, 622072, 622075, 606162, 621381,
            606131, 592879, 657041, 657571, 605156, 670329, 607231, 571697,␣
↪641312, 623214, 445055, 598271, 593334, 543135,
            282332, 594986, 444482, 641755, 605119, 592348, 571437, 656577,␣
↪572020, 452254, 608566, 650644, 541600, 592644,
            643327, 553878, 514913, 669456, 571670, 656887, 457915, 657566,␣
↪664247, 657145, 594840, 658792, 607200, 608385,
            656794, 448855, 502026, 663432, 456051, 542255, 596451, 592351,␣
↪669270, 621199, 519293, 502043, 572863, 456701,
            656222, 453172, 527048, 571918, 474463, 665487, 435079, 543118,␣
↪643493, 501985, 641627, 453562, 594311, 596119,
            643603, 554340, 656308, 622797, 455104, 607219, 663465, 502042,␣
↪668683, 592444, 621453, 642207, 669214, 660761,
            543089, 621249, 621141, 663776, 605135, 625643, 595956, 527049,␣
↪643354, 657077, 489334, 542921, 628452, 446321,
            664874, 592332, 608334, 621311, 606273, 621439, 592680, 501381,␣
↪607223, 608339, 602922, 621244, 630111, 543606,
            664045, 573135, 518452, 594987, 543699, 605276, 628356, 650671,␣
↪489119, 656977, 435064, 543432, 608344, 446263,
            468504, 641745, 542960, 650813, 642545, 607644, 608336, 570240,␣
↪605164, 592741, 670970, 448281, 448602, 657140,
            670950, 670456, 595375, 663531, 605508, 656252, 456488, 607776,␣
↪608717, 542963, 596001, 592620, 642098, 623352,
            605253, 643338, 664208, 640470, 656257, 623364, 471865, 453281,␣
↪571539, 596043, 613534, 446868, 664196, 660853,
              572140, 663898, 460576, 656288, 519008, 519076, 547982, 606424,␣
↪608638, 593528, 598286, 519455, 523260, 594011,
              596057, 434378, 570256, 676606, 571946, 642558, 657053, 595465,␣
↪623184, 445926, 642721, 624414, 434778, 595191,
              460026, 606959, 570632, 622766, 571745, 592229, 669203, 457705,␣
↪502239, 571510, 592717, 451192, 592468, 456665,
              605513, 672773, 429719, 605151, 641851, 657097, 605154, 456124,␣
↪596112, 543281, 546318, 596133, 446359, 605498,
              543001, 488768, 543883, 642397, 606149, 642073, 595453, 592387,␣
↪521655, 547170, 518553, 476451, 434538, 641941,
              543193, 450203, 663423, 609280, 502570, 572143, 624419, 543194,␣
↪605538, 598284, 669738, 593647, 622103, 544928,
              592716, 666198, 433589, 593423, 500208, 592865, 642003, 624407,␣
↪543776, 542881, 594988, 593700, 621559, 446372,
              547888, 621389, 607229, 457918, 570666, 571863, 640460, 622226,␣
↪491696, 621261, 488721, 621550, 592791, 657024,
              623454, 605520, 592685, 461314, 656775, 623167, 504379, 456030,␣
↪434670, 623149, 608715, 571595, 595885, 622554,
              642701, 592102, 658648, 593833, 643524, 458681, 543484, 600968,␣
↪650895, 656582, 630023, 598264, 519393, 584171,
              614173, 445276, 456034, 607192, 641438, 623515, 453064, 605439,␣
↪592779, 400085]
allAtBats = []
xPosList = []
zPosList = []
DescList = []
topSZ = []
botSZ = []
pType = []
Names = []
#Count_3_2 = []
i = 0

data2 = playerid_reverse_lookup(allPlayers, key_type='mlbam')
#print(data2)
allFirstNames = data2['name_first'].to_list()
allLastNames = data2['name_last'].to_list()
allPlayerID = data2['key_mlbam'].to_list()
allNames = {}
j = 0
for player in allFirstNames:
    nameTemp = player + " " + allLastNames[j]
    if nameTemp not in allNames:
        allNames[nameTemp] = allPlayerID[j]
    j+=1
```

```
#print(allNames)

k = 0
i = 0
for player in allNames:
    ID = allNames[player]
    for x in range(100000):                               # range(num_rows):␣
 ↪*****CHANGE BACK WHEN DOING FINAL
        tempID = (data.iloc[x]).iloc[6]
        if tempID == ID:
            Names.append(player)
            xPosList.append((data.iloc[x]).iloc[29])
            zPosList.append((data.iloc[x]).iloc[30])
            pType.append((data.iloc[x]).iloc[0])
            DescList.append((data.iloc[x]).iloc[9])
            topSZ.append((data.iloc[x]).iloc[50])
            botSZ.append((data.iloc[x]).iloc[51])
            atBat = [Names[i], xPosList[i], zPosList[i], pType[i], DescList[i]]
            allAtBats.append(atBat)
            i += 1
#### Classifications
classification = []
j = 0
for pitch in allAtBats:
    ## strike
    if (-0.71 < xPosList[j] < 0.71) and (botSZ[j] < zPosList[j] < topSZ[j]):
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
            classification.append("strike, good") # swung
        else:
            classification.append("strike, bad")  # didn't swing
    ## not strike
    else:
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
            classification.append("ball, bad")    # swung
        else:
            classification.append("ball, good")   # didn't swing
    allAtBats[j].append(classification[j])
    j += 1

ALLdf = pd.DataFrame(allAtBats, columns = ['Name', 'xPos', 'zPos', 'Pitch␣
 ↪Type', 'Given Description', 'Classification'])
print(ALLdf)
```

```
          ␣
↪---------------------------------------------------------------------------

       KeyboardInterrupt                         Traceback (most recent call␣
↪last)

       <ipython-input-38-5ed9946c5865> in <module>
       106     for x in range(100000):                              #␣
↪range(num_rows): *****CHANGE BACK WHEN DOING FINAL
       107         tempID = (data.iloc[x]).iloc[6]
   --> 108         if tempID == ID:
       109             Names.append(player)
       110             xPosList.append((data.iloc[x]).iloc[29])


       KeyboardInterrupt:
```

```
[37]: #### STATS FOR ALL OF MLB - Discernment Scores
League = []
for batter in allNames:
    num_pitches = 0
    num_strikes = 0
    num_balls = 0
    num_good_strikes = 0
    num_good_balls = 0
    for x in range(len(ALLdf)):
        if ((ALLdf.iloc[x]).iloc[0]) == batter:
            num_pitches += 1
            if ((ALLdf.iloc[x]).iloc[5]) == "strike, good" or ((ALLdf.iloc[x]).
↪iloc[5]) == "strike, bad":
                num_strikes += 1
                if ((ALLdf.iloc[x]).iloc[5]) == "strike, good":
                    num_good_strikes += 1
            else:
                num_balls += 1
                if ((ALLdf.iloc[x]).iloc[5]) == "ball, good":
                    num_good_balls += 1
    if num_strikes != 0 and num_balls != 0 and num_pitches >= 100:
        p_Good = num_good_strikes / num_strikes
        p_Bad = num_good_balls / num_balls
        LeagueDF.append([batter,round(num_pitches,3) ,round(num_strikes,3),␣
↪round(num_balls,3), round(p_Good,3),round(p_Bad,3),round((p_Good +␣
↪p_Bad),3)])
```

```
LeagueDF = pd.DataFrame(League, columns = ['Name','Pitches', 'Strikes',␣
↪'Balls', 'g_strike', 'g_ball', 'DS'])
print(LeagueDF)
```

```
Empty DataFrame
Columns: [Name, Pitches, Strikes, Balls, g_strike, g_ball, DS]
Index: []
```

[ ]:
```
#### NOTES/PRELIMINARY WORK
# z strike zone is 1.75 to 3.42 feet
# x strike zone is -0.71 to 0.71 feet
# top: 3.34, bottom: 1.57 ---> (-0.79, 1.57), 1, 1.77
# bottom left corner ->(x,y), width, height
# https://www.baseballprospectus.com/news/article/14098/
↪spinning-yarn-the-real-strike-zone-part-2/
# https://www.geeksforgeeks.org/different-ways-to-create-pandas-dataframe/
# https://stackoverflow.com/questions/17812978/
↪how-to-plot-two-columns-of-a-pandas-data-frame-using-points
# https://www.statology.org/matplotlib-rectangle/
# https://stackoverflow.com/questions/57246963/
↪why-isnt-the-legend-in-matplotlib-correctly-displaying-the-colors
# https://pythonexamples.org/pandas-dataframe-sort-by-column/#2
# https://stackoverflow.com/questions/8924173/
↪how-do-i-print-bold-text-in-python/8930747
# https://www.geeksforgeeks.org/change-figure-size-in-pandas-python/
# IMPORTANT: Cell -> Current Outputs -> Toggle Scrolling

#### all stats from year 2020
#stats = batting_stats(2020)
#print(stats)
#print("================================ AARON JUDGE␣
↪================================")
#### Player lookup
#print(playerid_lookup('Judge', 'Aaron'))
#print()

#### Data on player
#player_data = statcast_batter('2016-04-01', '2017-07-15', player_id = 592450)
#print(player_data)
#df, player_info_dict = get_splits('judgeaa01', player_info = True)
#print(df)
#print()

##### Turns homeruns column to a list
#HRList = df['HR'].to_list()
#print("Home runs in the last 365 days for Arron Judge: ", HRList[4])
```

```
#### Team batting stats for only 2019 season
#dataTeam = team_batting(2019)
#print(dataTeam)
#teams = team_ids(2019)
#batting = team_batting(2019).add_prefix('batting.')
#teams.merge(batting, left_on=['yearID', 'teamIDfg'], right_on=['batting.
 ↪Season', 'batting.teamIDfg'])

#print("================================ TEAM DATA␣
 ↪================================")
#dataNYY = team_batting_bref('NYY', 2019)
#print(dataNYY)
#all_hits = dataNYY['HR'].to_list()
#total = 0
#for num in all_hits:
#    total += int(num)
#print("Total home runs for NYY: ", total)
```

```
[ ]: #### FUNCTION FOR ALL TEAMS

     def teamData(teamName):
         dataTeam = team_batting_bref(teamName, 2019)
         players = dataTeam['Name'].to_list()
         listTeamPlayers = []     #### all team players




     for playerName in players:
         name = playerName.split()
         listPLayers.append(name)

     #### Stores players' ID in dict... Omits 2 players - no data
     playerNums = {}
     for player in listPLayers:
         playerName = player[0] + " " + player[1]
         playerLookUp = playerid_lookup(player[1], player[0])
         playerNum = playerLookUp['key_mlbam'].to_list()
         if len(playerNum) != 0:
             playerNums[playerName] = playerNum[0]    #### {'key': 'value'}

     #print(data.columns.get_loc('plate_x'))      #### finds index
     #print(data.columns.get_loc('plate_z'))
     #print(data.columns.get_loc('sz_top'))
     #print(data.columns.get_loc('sz_bot'))
     #print("Balls: ",data.columns.get_loc('balls'))  #24
     #print("Strikes: ",data.columns.get_loc('strikes'))  #25
```

```
index = data.index
num_rows = len(index)

YankAtBats = []      #### all yank pitches
xPosList = []        #### all x positions
zPosList = []        #### all z positions
DescList = []        #### all descriptions of pitches
topSZ = []           #### top of strike zone - predetermined
botSZ = []           #### bottom of strike zone - predetermined
pType = []           #### pitch type
Names = []           #### batter name
Count_3_2 = []
i = 0
gardyT = []
gardyB = []
for x in range(num_rows):
    ID = (data.iloc[x]).iloc[6]
    name = [key for key, v in playerNums.items() if v == ID]
    if name != [] and name[0] in playerNums:
        Names.append(name[0])
        xPosList.append((data.iloc[x]).iloc[29])
        zPosList.append((data.iloc[x]).iloc[30])
        pType.append((data.iloc[x]).iloc[0])
        DescList.append((data.iloc[x]).iloc[9])
        topSZ.append((data.iloc[x]).iloc[50])
        botSZ.append((data.iloc[x]).iloc[51])
        if ((data.iloc[x]).iloc[24]) == 3 and ((data.iloc[x]).iloc[25]) == 2:␣
 ↪#### pulls out all 3-2 counts
            Count_3_2.append(1)
        else:
            Count_3_2.append(0)
        if name[0] == "Brett Gardner":
            gardyT.append((data.iloc[x]).iloc[50])
            gardyB.append((data.iloc[x]).iloc[51])
        atBat = [Names[i], xPosList[i], zPosList[i], pType[i], DescList[i]]
        YankAtBats.append(atBat)
        i += 1

#### Classifications
classification = []
j = 0
for yankPitch in YankAtBats:
    ## strike
    if (-0.71 < xPosList[j] < 0.71) and (botSZ[j] < zPosList[j] < topSZ[j]):
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
```

```
            classification.append("strike, good") # swung
        else:
            classification.append("strike, bad")  # didn't swing
    ## not strike
    else:
        if DescList[j] == "hit_into_play" or DescList[j] == "swinging_strike"␣
 ↪or DescList[j] == "foul" or DescList[j] == "swinging_strike_blocked" or␣
 ↪DescList[j] == "foul_tip":
            classification.append("ball, bad")    # swung
        else:
            classification.append("ball, good")   # didn't swing
    YankAtBats[j].append(classification[j])
    j += 1

#### New data frame of all NYY pitches (size = 28512, last 53 bad)
newdf = pd.DataFrame(YankAtBats[0:len(YankAtBats)-53], columns = ['Name',␣
 ↪'xPos', 'zPos', 'Pitch Type', 'Given Description', 'Classification'])
print("FA - Fastball, CU - Curveball, FT - Two-seam Fastball, CH - Changeup,␣
 ↪\nFC - Cutter, SL - Slider, FS - Splitter, KN - Knuckleball,\nFF - Four-seam␣
 ↪Fastball \n")
print(newdf,"\n")
```