NBA ASSISTS: DATAFRAME PRACTICE

You are asked to make a dataframe of the following data; Rajon Rondo, Lebron James, James Harden, and Russell Westbrook's total assists, minutes per game, and total games played. You can get the data from http://www.espn.com/nba/statistics/player/_/stat/assists/sort/avgAssists/year/2018/seasontype/2

Let's get started.Enter your virtual environment and make your imports.

```
import pandas as pd
```

```
import numpy as np
```

THE POOR WAY

There are several ways we can structure the data. We can have our rows be the stats categories and our columns be the NBA players, or we have our columns be the NBA players and our columns be their chosen statistics (minutes per game, total games played, and total assists).

The way I recommend is the NBA players themselves be the rows, and the columns be the statistics. This is the common structure chosen. But first, let's do it the poor way.

Making Our Data Lists

```
russ = [80, 36.4, 820]
james = [82, 36.9, 747]
harden = [72, 35.4, 630]
rondo = [65, 26.2, 533]
```

Using Our Lists To Make Dictionaries

Let’s build the info. First we can make our data. Let’s make a list of each column. For now we will do it the poor way with the NBA players as our column, and our row labels being the statistics.Let’s build our NBA players. First, let’s create a list of each NBA’s players games played, minutes per game, assists total, in that order.

```
bad_data_dict = {'Russell Westbrook': russ, 'Lebron James': james, 'James Harden':
harden, 'Rajon Rondo': rondo}
```

Now let's print out our Dictionary to take a look.

```
bad_data_dict
```

```
{'Russell Westbrook': [80, 36.4, 820],
 'Lebron James': [82, 36.9, 747],
 'James Harden': [72, 35.4, 630],
 'Rajon Rondo': [65, 26.2, 533]}
```

As you can see our dictionary is simply 4 entries. Each entry has a list that holds 3 values. We can think of this having a similar data structure as a 4 x 3 array.

Making Our Labels List

Next we need to make a list comprising our labels.

```
bad_labels = ['Games Played', 'Minutes Per Game', 'Total Assists']
```

Making Our Dataframe

Now that we have our data dictionary and our labels, we can make our Dataframe by passing our dictionary and labels list as arguments to our pd.Dataframe() method.

```
bad_df = pd.DataFrame(bad_data_dict, index=bad_labels)
```

```
bad_df
```

.dataframe tbody tr th:only-of-type { vertical-align: middle; } .dataframe tbody tr th { vertical-align: top; } .dataframe thead th { text-align: right; }

	James Harden	Lebron James	Rajon Rondo	Russell Westbrook
Games Played	72.0	82.0	65.0	80.0
Minutes Per Game	35.4	36.9	26.2	36.4
Total Assists	630.0	747.0	533.0	820.0

When we print our dataframe we can see that our column names our the player names, and our row labels are the names of hte statistics we are gathering.

This is bad form. The better way to do it is having the names of categories for the statistics you're collecting as column names, and each instance, or record for such statistics as the labels.

Here, the statistics we're looking for are Games Played, Minutes Per Game, and Assists Per Game. And each record for such statistics are the NBA Players who recorded such statistics. Therefore we should have the NBA Player names be the row labels and the category names be the Column names.

The Good Way

Making Our Data Lists

Let's recreate our dataframe. This time we will use the NBA player names as labels and the category names as our Column names.

To do this, let's create a list of each NBA player's game's played, minutes per game, and total assists, making sure that we order the lists by the name players, i.e., the first number referring to Russel Westbrook, the second number referring to Lebron James, the third number referring to James Harden, and the second number referring to Rajon Randon.

```
gp = [80, 82, 72, 65]
mpg = [36.4, 36.9, 35.4, 26.2]
ast = [820, 747, 630, 533]
```

Using Our Lists To Make Dictionaries

Now let's make our dictionary as before, this time with the dictionary keys being the category names, and the values being the lists of values for each respective category.

```
data_dict = {'Total Assists': ast, 'Games Played': gp, 'Minutes Per Game': mpg}
```

```
data_dict
```

```
{'Total Assists': [820, 747, 630, 533],
 'Games Played': [80, 82, 72, 65],
 'Minutes Per Game': [36.4, 36.9, 35.4, 26.2]}
```

Making Our Labels List

This time our labels will be the player names themselves, as opposed to the category names.

```
labels = ['Russell Westbrook', 'Lebron James', 'James Harden', 'Rajon Randon']
```

Making Our Dataframe

```
df = pd.DataFrame(data_dict, labels)
```

```
df
```

.dataframe tbody tr th:only-of-type { vertical-align: middle; } .dataframe tbody tr th { vertical-align: top; } .dataframe thead th { text-align: right; }

	Games Played	Minutes Per Game	Total Assists
Russell Westbrook	80	36.4	820
Lebron James	82	36.9	747
James Harden	72	35.4	630
Rajon Randon	65	26.2	533

Selecting A Column

Let's select the Assist column just to make sure we know how to select columns.

```
df['Total Assists']
```

```
Russell Westbrook    820
Lebron James         747
James Harden         630
Rajon Randon         533
Name: Total Assists, dtype: int64
```

Creating New Dataframe Columns By Combining Other Columns

We can also combine columns of our dataframe to create new dataframe colulmns.

Creating Assists Per Game Column

For instance, if we want to get the total amount of Assists per game, we can simply divided each players total assists by the total amount of games played.

```
df['Assists Per Game'] = df['Total Assists'] / df['Games Played']
```

```
df
```

.dataframe tbody tr th:only-of-type { vertical-align: middle; } .dataframe tbody tr th { vertical-align: top; } .dataframe thead th { text-align: right; }

	Games Played	Minutes Per Game	Total Assists	Assists Per Game
Russell Westbrook	80	36.4	820	10.250000
Lebron James	82	36.9	747	9.109756
James Harden	72	35.4	630	8.750000
Rajon Randon	65	26.2	533	8.200000

Creating Total Minutes Column

Let's create total minutes by multiplying minutes per game with games played.

```
df['Total Minutes'] = df['Minutes Per Game'] * df['Games Played']
```

Creating Assists Per Minute Column

Now we can create a total assists per minute column by dividing total assists by total minutes.

```
df['Assists Per Minute'] = df['Total Assists'] / df['Total Minutes']
```

```
df
```

.dataframe tbody tr th:only-of-type { vertical-align: middle; } .dataframe tbody tr th { vertical-align: top; } .dataframe thead th { text-align: right; }

	Games Played	Minutes Per Game	Total Assists	Assists Per Game	Total Minutes	Assists Per Minute
Russell Westbrook	80	36.4	820	10.250000	2912.0	0.281593
Lebron James	82	36.9	747	9.109756	3025.8	0.246877
James Harden	72	35.4	630	8.750000	2548.8	0.247175
Rajon Randon	65	26.2	533	8.200000	1703.0	0.312977

Filtering Data: Returning Data For Those Players Who Played More than 2000 Minutes

Lastly, we can filter our data. Let's say we only want to print out the data for those players who played more than 2000 minutes. Here's how we would do it.

```
df[df['Total Minutes'] > 2000]
```

.dataframe tbody tr th:only-of-type { vertical-align: middle; } .dataframe tbody tr th { vertical-align: top; } .dataframe thead th { text-align: right; }

	Games Played	Minutes Per Game	Total Assists	Assists Per Game	Total Minutes	Assists Per Minute
Russell Westbrook	80	36.4	820	10.250000	2912.0	0.281593
Lebron James	82	36.9	747	9.109756	3025.8	0.246877
James Harden	72	35.4	630	8.750000	2548.8	0.247175

Selecting A Row

And if we want to see the data for only one player, aka one row, we can simply use .loc[]

```
df.loc['Rajon Randon']
```

```
Games Played            65.000000
Minutes Per Game        26.200000
```

```
Total Assists          533.000000
Assists Per Game         8.200000
Total Minutes         1703.000000
Assists Per Minute       0.312977
Name: Rajon Randon, dtype: float64
```