Application Programming Interface

Estimated time needed: **15** minutes

Objectives

After completing this lab you will be able to:

- Create and Use APIs in Python

Introduction

An API lets two pieces of software talk to each other. Just like a function, you don't have to know how the API works, only its inputs and outputs. An essential type of API is a REST API that allows you to access resources via the internet. In this lab, we will review the Pandas Library in the context of an API, we will also review a basic REST API

Table of Contents

You will use this function in the lab:

```python
def one_dict(list_dict):
    keys=list_dict[0].keys()
    out_dict={key:[] for key in keys}
    for dict_ in list_dict:
        for key, value in dict_.items():
            out_dict[key].append(value)
    return out_dict
```

Pandas is actually set of software components , much of which is not even written in Python.

```python
import pandas as pd
import matplotlib.pyplot as plt
```

You create a dictionary, this is just data.

```python
dict_={'a':[11,21,31],'b':[12,22,32]}
```

When you create a Pandas object with the dataframe constructor, in API lingo this is an "instance". The data in the dictionary is passed along to the pandas API. You then use the dataframe to communicate with the API.

```python
df=pd.DataFrame(dict_)
type(df)
```

When you call the method `head` the dataframe communicates with the API displaying the first few rows of the dataframe.

```
df.head()
```

When you call the method `mean`, the API will calculate the mean and return the value.

```
df.mean()
```

It's quite simple to use the nba api to make a request for a specific team. We don't require a JSON, all we require is an id. This information is stored locally in the API. We import the module `teams`

```
!pip install nba_api

from nba_api.stats.static import teams
import matplotlib.pyplot as plt

#https://pypi.org/project/nba-api/
```

The method get_teams() returns a list of dictionaries.

```
nba_teams = teams.get_teams()
```

The dictionary key id has a unique identifier for each team as a value. Let's look at the first three elements of the list:

```
nba_teams[0:3]
```

To make things easier, we can convert the dictionary to a table. First, we use the function one dict, to create a dictionary. We use the common keys for each team as the keys, the value is a list; each element of the list corresponds to the values for each team. We then convert the dictionary to a dataframe, each row contains the information for a different team.

```
dict_nba_team=one_dict(nba_teams)
df_teams=pd.DataFrame(dict_nba_team)
df_teams.head()
```

Will use the team's nickname to find the unique id, we can see the row that contains the warriors by using the column nickname as follows:

```
df_warriors=df_teams[df_teams['nickname']=='Warriors']
df_warriors
```

we can use the following line of code to access the first column of the dataframe:

```
id_warriors=df_warriors[['id']].values[0][0]
# we now have an integer that can be used to request the Warriors
information
id_warriors
```

The function "League Game Finder " will make an API call, it's in the module stats.endpoints

```
from nba_api.stats.endpoints import leaguegamefinder
```

The parameter team_id_nullable is the unique ID for the warriors. Under the hood, the NBA API is making a HTTP request. The information requested is provided and is transmitted via an HTTP response this is assigned to the object game finder.

```
# Since https://stats.nba.com does not allow api calls from Cloud IPs
and Skills Network Labs uses a Cloud IP.
# The following code is commented out, you can run it on jupyter labs
on your own computer.
# gamefinder =
leaguegamefinder.LeagueGameFinder(team_id_nullable=id_warriors)
```

we can see the json file by running the following line of code.

```
# Since https://stats.nba.com does not allow api calls from Cloud IPs
and Skills Network Labs uses a Cloud IP.
# The following code is commented out, you can run it on jupyter labs
on your own computer.
# gamefinder.get_json()
```

The game finder object has a method get_data_frames(), that returns a dataframe. If we view the dataframe, we can see it contains information about all the games the Warriors played. The PLUS_MINUS column contains information on the score, if the value is negative, the Warriors lost by that many points, if the value is positive, the warriors won by that amount of points. The column MATCHUP has the team the Warriors were playing, GSW stands for Golden State Warriors and TOR means Toronto Raptors. vs signifies it was a home game and the @ symbol means an away game.

```
# Since https://stats.nba.com does not allow api calls from Cloud IPs
and Skills Network Labs uses a Cloud IP.
# The following code is comment out, you can run it on jupyter labs on
your own computer.
# games = gamefinder.get_data_frames()[0]
# games.head()
```

you can download the dataframe from the API call for Golden State and run the rest like a video.

```
import requests
```

```
filename = "https://s3-api.us-geo.objectstorage.softlayer.net/cf-
courses-data/CognitiveClass/PY0101EN/Chapter%205/Labs/
Golden_State.pkl"

def download(url, filename):
    response = requests.get(url)
    if response.status_code == 200:
        with open(filename, "wb") as f:
            f.write(response.content)

download(filename, "Golden_State.pkl")

file_name = "Golden_State.pkl"
games = pd.read_pickle(file_name)
games.head()
```

We can create two dataframes, one for the games that the Warriors faced the raptors at home, and the second for away games.

```
games_home=games[games['MATCHUP']=='GSW vs. TOR']
games_away=games[games['MATCHUP']=='GSW @ TOR']
```

We can calculate the mean for the column PLUS_MINUS for the dataframes games_home and games_away:

```
games_home['PLUS_MINUS'].mean()

games_away['PLUS_MINUS'].mean()
```

We can plot out the PLUS MINUS column for the dataframes games_home and games_away. We see the warriors played better at home.

```
fig, ax = plt.subplots()

games_away.plot(x='GAME_DATE',y='PLUS_MINUS', ax=ax)
games_home.plot(x='GAME_DATE',y='PLUS_MINUS', ax=ax)
ax.legend(["away", "home"])
plt.show()
```

Calculate the mean for the column PTS for the dataframes games_home and games_away:

```
# Write your code below and press Shift+Enter to execute
```

Authors:

Joseph Santarcangelo

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-09-09	2.1	Malika Singla	Spell Check
2020-08-26	2.0	Lavanya	Moved lab to course repo in GitLab