# Weather API Script – PANDAS

# Dependencies and Setup

import matplotlib.pyplot as plt

import pandas as pd

import numpy as np

import requests

import time

# Output File (CSV)

%cd "C:\Users\kwlei\Documents\GitHub\WeatherReport"

output\_data\_file = "output\_data/cities.csv"

# Import API key

from api\_key import api\_key

from citipy import citipy

# Range of latitudes and longitudes

# lat\_range = (-58, 73)

# lng\_range = (-180, 180)

# List for holding lat\_lngs and cities

lat\_lngs = []

cities = []

# Create a set of random lat and lng combinations

lats = np.random.uniform(low=-90.000, high=90.000, size=1500)

lngs = np.random.uniform(low=-180.000, high=180.000, size=1500)

lat\_lngs = zip(lats, lngs)

# Identify nearest city for each lat, lng combination

for lat\_lng in lat\_lngs:

city = citipy.nearest\_city(lat\_lng[0], lat\_lng[1]).city\_name

# If the city is unique, then add it to a our cities list

if city not in cities:

cities.append(city)

cityGrab\_df = pd.DataFrame(cities, columns=["NAME"])

cityGrab\_df.to\_csv('cityGrab.csv', index=False)

# Print the city count

len(cities)

# Starting URL for Weather Map API Call

api\_key = "7bedea6925812000099cef9c945868ec"

url = "http://api.openweathermap.org/data/2.5/weather?units=Imperial&APPID=" + api\_key

# List of city data

city\_data = []

# Print to logger

print("Start Data")

print("-----------------")

# Create counters

record\_count = 1

set\_count = 1

# Loop through all the cities in our list

for i, city in enumerate(cities):

# Group cities in sets of 50 for logging purposes

if (i % 50 == 0 and i >= 50):

set\_count += 1

record\_count = 0

# Create endpoint URL with each city

city\_url = url + "&q=" + city

# Log the url, record, and set numbers

print("Processing Record %s of Set %s | %s" % (record\_count, set\_count, city))

# Add 1 to the record count

record\_count += 1

# Run an API request for each of the cities

try:

# Parse the JSON and retrieve data

city\_weather = requests.get(city\_url).json()

# Parse out the max temp, humidity, and cloudiness

city\_lat = city\_weather["coord"]["lat"]

city\_lng = city\_weather["coord"]["lon"]

city\_max\_temp = city\_weather["main"]["temp\_max"]

city\_humidity = city\_weather["main"]["humidity"]

city\_clouds = city\_weather["clouds"]["all"]

city\_wind = city\_weather["wind"]["speed"]

city\_country = city\_weather["sys"]["country"]

city\_date = city\_weather["dt"]

# Append the City information into city\_data list

city\_data.append({"City": city,

"Lat": city\_lat,

"Lng": city\_lng,

"Max Temp": city\_max\_temp,

"Humidity": city\_humidity,

"Cloudiness": city\_clouds,

"Wind Speed": city\_wind,

"Country": city\_country,

"Date": city\_date})

# If an error is experienced, skip the city

except:

print("City not found. Skipping...")

pass

# Indicate that Data Loading is complete

print("-----------------------------")

print("Data Retrieval Complete ")

print("-----------------------------")

city\_data\_pd = pd.DataFrame(city\_data)

city\_weather9\_df = pd.DataFrame(city\_data)

city\_weather9\_df.to\_csv('CityWeather9.csv', index=False)

# lats = city\_data\_pd["Lat"]

city\_weather9\_df.tail()