Universidad de Sonora
Departamento de Física
Licenciatura en Física
Computacional I
2017-1

Reporte

Actividad #3: Python y limpieza de datos

Leonardo Coronado Arvayo
Profesor: Carlos Lizárraga Celaya

13 de febrero de 2017

Índice

Resumen

La actividad 3 trata sobre la limpieza de datos y su manejo usando Python. En especifico se obtuvieron las observaciones disponibles de la Energía Potencial Conectiva Disponible (CAPE, por sus siglas en ingles) y la precipitación del agua (PW) para la estación meteorológica de la Paz, Baja California en el 2016, y se extrajo información de estas (visual y gráfica) usando Python.

1. Introducción

Primero que nada se considera relevante definir brevemente las variables alrededor de las cuales se centrara este trabajo que son:

- CAPE es "... la cantidad de energía que un parcel de aire tendría si se levantara una cierta distancia verticalmente a través de la atmósfera"[1].
- PW es "... es la profundidad del agua en una columna de la atmósfera, si esta fuera precipitada como la lluvia, su unidad es en milímetros o pulgadas"[2].

CAPE es un indicador de inestabilidad atmosférica, que es util para predecir clima severo[1]. De acuerdo a [2] PW se puede usar para calcular indirectamente la humedad.
En este trabajo se obtendrán las observaciones que existe de estas de variables para el año 2016, se limpian y se procesan sencillamente con Python.

2. Desarrollo

2.1. Limpieza de datos

Para la limpieza de datos primeramente se uso Linux/Bash, primero se extrajo las variables de interés del archivo "sondeos.txt."en el cual se tienen todas las variables que el sitio de la Universidad de Wyoming[3]. Para esto se uso el comando (en la terminal de Linux):

```
cat sondeos.txt |egrep -i "Observations|CAPE|precipitable"| sed -i "00Z,2d" > 12Z.txt
```

El cual se encarga primero de considerar el archivo sondeos.txt, luego ubicar las palabras ."ºbservations"(para la fecha), ÇAPE"para CAPE y "precipitable"para PW, note que espacios y mayúsculas cuentan. La ultima parte se encarga de filtrar todos los valores que no tengan la 00Z horas de Greenwich y las pone en un archivo, en este caso 12Z por que es la hora de interés.

Luego se procede a buscar si existen otras horas que no son las 12Z, esto se hace a través de:

```
grep 00Z 12Zcape.txt
grep 08Z 12Zcape.txt
grep 09Z 12Zcape.txt
```

Que se encarga de buscar valores distintos a las 12Z, pero no se encontró ninguno.
Una vez que el archivo contenía solo la hora y variables de interés, se paso a usar emacs para eliminar texto sobrantes. Esto se hizo a través del comando "query-replace."al cual se accede aplastando

.[Es]c-x.[en] emacs y luego escribiendo "query-replace".

El proceso es el siguiente, primero se selecciona el texto que se desea eliminar, preferentemente uno que se repita en el texto, se copea con la interfaz o con .[A]lt-w". Luego se accede a la terminal de emacs .[Es]c-xz se escribe "query-replace", se aplasta .[en]ter". Una ver hecho esto te aparece el texto que se desea elimanar, se selecciona pegando con la interfaz o con Çtrl-y", se aplasta enter y te pregunta emacs si deseas sustituirlo con algo, en caso de requerirlo se escribe y se aplasta enter. Por ultimo se confirma la orden con el signo ¡".

Se repite este proceso tres veces hasta que los datos queden como se ve la siguiente tabla:

Fecha,	CAPE,	PW
01 Jan 2016,	0,	21.79
02 Jan 2016,	0,	23.06
03 Jan 2016,	0,	17.56
04 Jan 2016,	2.18,	17.32
06 Jan 2016,	0,	19.05
⋮,	⋮,	⋮

Tabla 1: Estructura de los datos

Note que se incluyo una coma y un espacio entre los datos.

2.2. Comienzos con Python

Una vez que se obtuvieron solo los datos CAPE y PW con sus fechas correspondientes se procedió a nombrar al archivo con una extensión çsv", el cual es usado en Python. Primero se le dijo a Python las librerías necesarias, a través del siguiente comando:

```
import pandas as pd
import numpy as np
import matplotlib as plt
```

Seguido, se le dijo a Python donde se ubica el archivo en cuestión, que para este caso fue:

```
df = pd.read_csv ("/home/.../12Z.csv")
```

Para comprobar que el archivo es el correcto se da el siguiente comando:

```
df.head(20)
```

Que muestra las primeras 20 observaciones, el resultado fue:

	Fecha	CAPE	PW
0	01 Jan 2016	0.00	21.79
1	02 Jan 2016	0.00	23.06
2	03 Jan 2016	0.00	17.56
3	04 Jan 2016	2.18	17.32
4	06 Jan 2016	0.00	19.05
5	08 Jan 2016	0.00	9.99
6	09 Jan 2016	0.00	13.23
7	10 Jan 2016	0.00	10.64
8	11 Jan 2016	0.00	14.11
9	12 Jan 2016	0.00	14.08
10	13 Jan 2016	0.00	13.20
11	14 Jan 2016	0.00	9.57
12	20 Jan 2016	0.00	9.15
13	21 Jan 2016	0.00	8.97
14	22 Jan 2016	0.00	8.65
15	24 Jan 2016	0.00	8.78
16	25 Jan 2016	0.00	14.17
17	26 Jan 2016	0.00	17.46
18	27 Jan 2016	0.00	10.20
19	28 Jan 2016	0.00	26.22

Figura 1: Archivo

Luego se aplico un comando para describir las variables en cuestion, a traves del comando:

```
df.describe()
```

	CAPE	PW
count	199.000000	199.000000
mean	27.510553	26.743266
std	109.972963	14.913804
min	0.000000	6.210000
25%	0.000000	13.535000
50%	0.000000	23.840000
75%	0.000000	39.060000
max	757.630000	76.190000

Figura 2: Indicadores para CAPE y PW

Cabe denotar que la variable CAPE presento muchas observaciones en 0. Luego se procedió a limpiar los datos, primero intentando ubicar el numero de archivos "NA.º sin valores, mediante:

```
df.apply(lambda x: sum(x.isnull()),axis=0)
```

El resultado fue 0 observaciones NA, es decir, que de las 199 valores de cada variable, todos eran medidas tomadas. Se aplico el comando para limpiar pero, obviamente, no hubo cambios. Seguido se paso a graficar, comenzando con el siguiente comando:

```
%matplotlib inline
```

Que permite mostrar las gráficas en el mismo archivo de Python, de otra manera se harían en otra ventana. Luego se pasa a preguntar por el nombre de las variables con:

```
df.columns
```

El resultado fue "Fecha", ÇAPE", "PW", es decir que las variables tienen un espacio antes de su nombre. Pero dado que no se considero una molestia significante. Procedió el proceso de graficación a través de:

```
df_clean[' CAPE'].hist(bins=100)
```

El resultado fue el siguiente histograma para CAPE, donde se puede observar que la variable esta fuertemente distribuida en el valor de 0. Este histograma es observable en la siguiente figura:

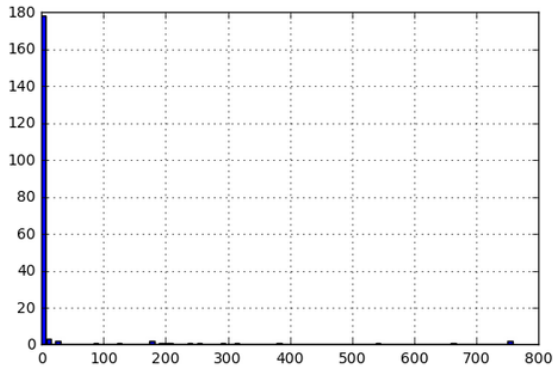

Figura 3: Histograma de CAPE

Luego se paso a hacer un diagrama de caja con el siguiente comando:

```
df.boxplot(column=' CAPE', return_type='axes')
```

Resultando en la siguiente figura:

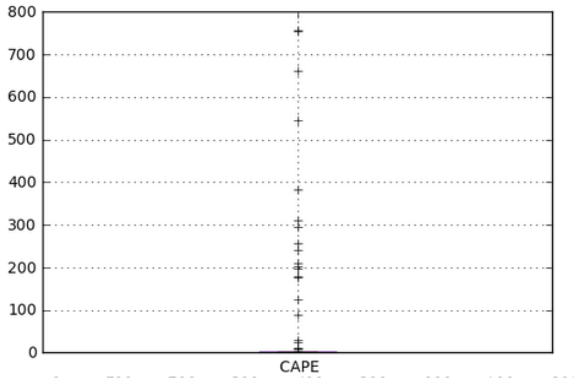

Figura 4: Diagrama de caja de CAPE

Seguido se procedió a intentar graficar un diagrama de caja para cada mes del año, para ello primero se creo una variable de mes con:

```
df['month']=pd.DatetimeIndex(df[' Fecha']).month
```

Definiendo a enero como 1, febrero como 2, y así sucesivamente para los demás meses. Luego se procedio a realizar la gráfica con:

```
df.boxplot(column=' CAPE', by= 'month', return_type='axes')
```

Que resulto en:

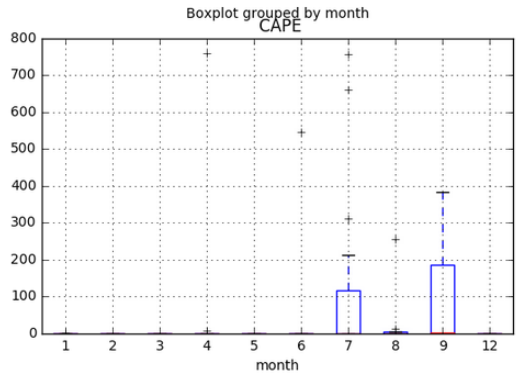

Figura 5: Diagrama de caja por mes de CAPE

Luego se procedió a realizar los mismos comando para PW, que al no tener tantos valores en 0, resulto en gráficas mas cambiantes. Como se puede observar a continuación (ya que se repitieron los comandos se agruparan en el siguiente párrafo.

```
df_clean[' PW'].hist(bins=100)
df.boxplot(column=' PW', return_type='axes')
df.boxplot(column=' PW', by= 'month', return_type='axes')
```

El histograma se observa en la siguiente figura:

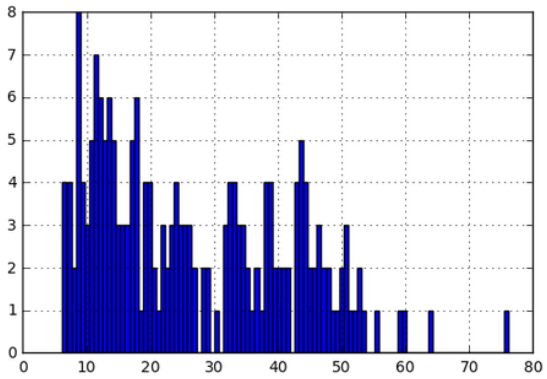

Figura 6: Histograma de PW

Como se puede observar las observaciones de PW no están concentradas como en el caso de CAPE, hecho que se reafirma en las siguientes gráficas.

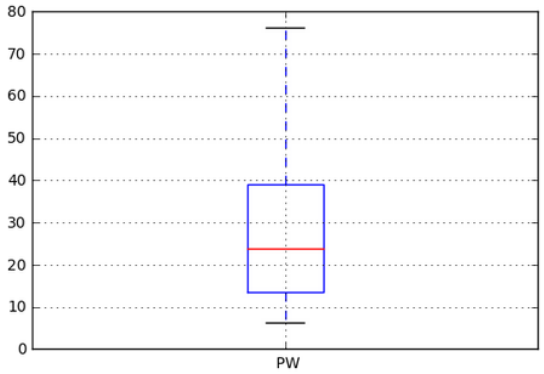

Figura 7: Diagrama de caja de PW

Tanto el histograma por mes como el anual siguieren un valor medio de PW de alrededor de 25, ademas, es interesante ver los cambios mensuales de la misma que parecen tener un comportamiento ondulatorio.

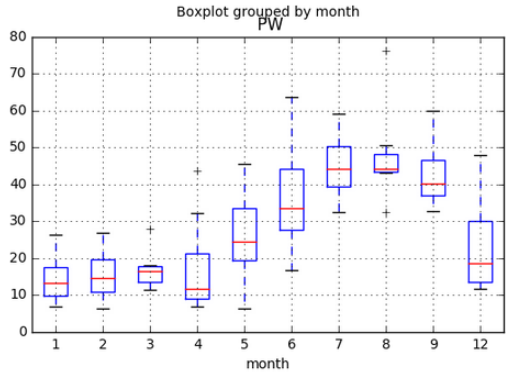

Figura 8: Diagrama de caja por mes de PW

3. Conclusiones

Primero que nada el uso de bash/linux para la extracción de datos resulto bastante útil, en especial cuando se considera la magnitud de los datos que se ignoran. Permitiendo concentrarse solo en las variables de interés.
También el uso de emacs para la limpieza de datos resulto bastante efectivo, ya que permite quitar valores repetido que no son necesarios para el presente estudio.
Por ultimo Python resulto mucho mas sencillo de usar de lo esperado, aunque bien es solo un comienzo, es bastante claro.

Referencias

[1] Convective available potential energy, Wikipedia: `https://en.wikipedia.org/wiki/Convective_available_potential_energy`

[2] Precipitable water. Wikipedia: https://en.wikipedia.org/wiki/Precipitable_water

[3] Departament of Atmospheric Science, University of Wyoming http://weather.uwyo.edu/upperair/sounding.html