CENTRO DE ENSEÑANZA TÉCNICA Y SUPERIOR

Escuela de Ingeniería en Ciencias Computacionales

Ingeniería en Ciencias Computacionales

Materia: Aprendizaje De Máquina

Presenta:

Pablo Díaz 030343

Kevin Huerta 30502

Gerardo Hernández 29902

Diego Garibay 30046

Profesor: Ulises Orozco Rosas

Practica #2: Introducción

Tijuana, B.C., 9 de septiembre de 2020

Aprendizaje Maquina Practica 2

Gerardo Hernandez 29902 [1], Pablo Diaz 30343[2], Kevin Huerta 30502[3] y Alejandro Zuazo 30046[4]

***Abstract*— En este reporte mencionaremos como fue el proceso que tomó hacer la practica 2 de la materia Aprendizaje de máquina. Anexando los conocimientos principales requeridos para la actividad, siendo fundamentales para darle solución a los problemas que conllevaba.**

I. INTRODUCCION

Este reporte tiene como objetivo explicar las metodologías aplicadas en los ejercicios resueltos de la practica número 2, a través del uso de dos lenguajes de programación: Python y R. Donde el enfoque principal que tiene esta práctica fue hacer uso de matrices, vectores y tanto como la implementación de gráficas en dos dimensiones y tres dimensiones en ambos lenguajes. Antes de entrar a la metodología, explicaremos algunos fundamentos que tuvimos que tener en cuenta para desarrollar nuestros algoritmos de resolución. Será una pirática complicada para python, ya que los ejercicios propuestos están diseñados para el lenguaje de R, pero investigando podemos encontrar algunas librerías que hagan algunas funciones que tiene integrado R.

II. FUNDAMENTOS

A. Declaración de Arreglos

Para Python necesitaremos importar una librería llamada Numpy que sirve para gestionar de mejor manera vectores y matrices en caso de necesitarlas. Esta librería nos servirá mucho para ahorrar líneas de código y que nuestro archivo sea más compacto. Entonces, para declarar arreglos de la librería numpy se hace: nombre de la variable seguido de "=" np.array([elemento1, elemento2, elemento3, etc]). Ejemplos:

- x1 = np.array([43,44,45,46,47])
- y1 = np.array([41,45,49,47,44])

En R se facilita la declaración de arreglos. Para declarar arreglos se usa la función c, que viene de combine, y los elementos dentro del paréntesis deben estar separados por comas. Los valores de los arreglos pueden ser booleanos, numéricos o cadenas. Estos son algunos ejemplos: [**?**]

- vectornumerico ← c(1, 3, 5, 7) #vector numérico
- vectortexto ← c("a", "b", "c", "d") #vector de texto
- vectorlogico ← c(TRUE, FALSE, TRUE) #vector lógico / booleano

B. Ciclos

Los ciclos en Python tienen una sintaxis un tanto diferente de lenguajes como C, Java,etc. La sintaxis es la siguiente:

```
for i in range(1,10):
    Code a ejecutar
```

Donde se utiliza la función range para recorrer una serie de números, el cual puede recibir 1 o 2 parámetros. Cuando se utiliza 1 parámetro, el for se repite desde el 0 hasta el parámetro recibido. Cuando son 2 parámetros, recorre desde el primer parámetro hasta el segundo.
Ahora, los ciclos en R tienen una sintaxis muy parecida a python,y su sintaxis es la siguiente:

```
for(<variable> in <objeto iterable>) <codigo>
```

Cuando se tiene un vector, se recorrería de la siguiente manera:

```
x <- c("a", "b", "c", "d")
for(i in seq_along(x)) {
    print(x[i])
}
```

Se usa la función seq_along la cual genera una secuencia numérica que usaremos para indicar los índices de los elementos de los objetos que queremos recorrer.[**?**]

C. Operaciones con Vectores

Primordialemente, antes de programar, fue indispensable saber la teoria elemental para efectuar operaciones con vectores. Los fundamentos indispensables para entender las operaciones con vectores que se manejo en ambos lenguajes, fueron:

- Suma de dos vectores.

$$x + z \doteq [x_1 + z_1, x_2 + z_2, \ldots, x_m + z_m]$$

- Resta de dos vectores.

$$x - z \doteq [x_1 - z_1, x_2 = z_2, \ldots, x_m - z_m]$$

- Mutilplicacion de un vector por escalar.

$$xc \doteq [x_1 - z_1, x_2 - z_2, \ldots, x_m - z_m]$$

- Producto punto entre dos vectores

$$w \cdot x \doteq \Sigma_{i=1}^{m} w_i \cdot x_i$$

- Producto punto entre una matriz y un vector

$$W \cdot x = \begin{bmatrix} w^{(1,1)} & w^{(1,2)} \\ w^{(2,1)} & w^{(2,2)} \end{bmatrix} \begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix}$$

$$\doteq \begin{bmatrix} w^{(1,1)}x^{(1)} + w^{(1,2)}x^{(2)} \\ w^{(2,1)}x^{(1)} + w^{(2,2)}x^{(2)} \end{bmatrix}$$

D. Gráficas

Para la implementación de las gráficas en R, utilizamos la funciona **plot()** que de acuerdo con la documentación de R, es una función genérica que tiene múltiples métodos dentro de ella. La cual, dependiendo del tipo de objeto que se este pasando a **plot()**, se llamara respectivamente cierto método interno. Como argumentos, tiene la capacidad de aceptar vectores, funciones, matrices. Otorgando un control en los aspectos del diseño de la misma gráfica. Además la función **plot()** puede mostrar gráficas tanto en dos dimensiones como tres dimensiones.

Para la implementación de las gráficas en Python, hicimos uso de una librería llamada Plotly....etc...etc

III. METODOLOGÍA

La metodología fue sencilla y práctica, nos dividimos en dos grupos para la implementación del código, un grupo de R y un grupo de Python. Nos cambiaremos de grupo en cada práctica para tratar de manejar los dos lenguajes de programación. Cada ejercicio tuvo su nivel de dificultad, pero determinamos que aún son problemas de acondicionamiento para familiarizarnos más con este nuevo lenguaje de programación como lo es R.

A. Ejercicio número 1

En el ejercicio número 1 se hizo una declaración de un arreglo de 1 dimensión, por medio de la función **c()** de R. Después se imprimió el arreglo simplemente escribiendo la variable a la que fue asignado el arreglo, en este caso x.

Después, se asigno de nuevo a x un nuevo arreglo de 1 dimensión, de nuevo con el comando **c()**, y creamos un nuevo arreglo asignado a y. Posteriormente imprimimos la longitud del arreglo con la función **length()**.

Más adelante, usamos la función **ls()** para mostrar las variables declaradas y usamos **rm()** para eliminar dichas variables. De hecho, probamos eliminar todas las variables usando **ls()** y **rm()**.

Seguidamente, probamos la función **matrix()** que prácticamente crea una matriz de números. Creamos una matriz asignada a x donde se especificó el número de filas y de columnas. Y usamos la función **sqrt()** a la matriz que saca la raíz cuadrada a los valores.

Para terminar usamos una función **rnorm()** que crea un arreglo dominada por una distribución normal. También usamos una función **set.seed()** que genera una secuencia de números aleatorios. Después se uso la función **var()** para calcular la varianza.

B. Ejercicio número 2

En el ejercicio numero 2 se hizo uso de la función **plot()** por parte de R, con la cual se graficaron dos gráficas. Por default se implemento un diagrama de dispersión. Usando como datos de entrada, 100 datos arbitrarios con la función **rnorm()**. Además también implementamos gráficas en tres dimensiones por lo cual hicimos uso de la función **persp()**.

C. Ejercicio número 3

En este ejercicio utilizamos la función **plot()** para graficar, esta función ya está incorporada en R por lo tanto no fue un problema, se utilizaron los vectores ya hechos x - y como los datos de la gráfica. Después se guardo la gráfica en un PDF con la función **pdf()**. Posteriormente usamos la función **seq()** para crear una secuencia, y la imprimimos. Ahora, usamos la función **contour()** para dibujar líneas en un gráfico 2D. También se utilizó la función **outer()** que crea la multiplicación de dos arreglos. Por último se uso la función **persp()** que sirve para crear gráficos avanzados en 3D.

D. Ejercicio número 4

El ejercicio numero 4 tuvo como objetivo hacer lectura al formato csv (comma separated values) con el propósito de visualizar para posteriormente hacer uso de dicha informacion. Por lo tanto hicimos uso de 2 funciones que nos permitieron hacer dicha lectura. **read.csv() y read.table()**

E. Ejercicio número 5

Este ejercicio se basó más en la creación de gráficas de otro tipo. De hecho se uso la misma función **plot()** para gráficar, lo único que cambió fueron los parámetros, donde el primer parámetro se refiere a que tipo de gráfica se quiere mostrar. Se uso la función **hist()** también que sirve para crear histograma. Posteriormente se utilizó la función **pairs()** para crear u diagrama de dispersión por cada par de variables en el argumento. La función **indetify()** lee la posición del puntero gráfico cuando se presiona el primer botón del mouse. La función **summary()** produce un resumen numérico de cada variable en un conjunto de datos (usado en gráficas mayormente).

F. Ejercicio número 6

En este punto efectuamos operaciones con vectores, por lo cual consideramos en usar la librería de python llamada numpy. Donde dicha librería nos ayudo a efectuar operaciones con vectores de una manera eficaz, óptima y sencilla.

G. Ejercicio número 7

En python se hizo uso de la libreria matplotlib para los diagramas de dispersión **matplotlib.pyplot.scatter()**. Ademas para los datos arbitrarios se hizo uso de numpy **numpy.random.normal()**. Y para la graficas en tres dimensiones de igual forma se uso la libreria matplotlib con la funcion **matplotlib.pyplot.figure().axes().contour3D()**

H. Ejercicio número 8

En este ejercicio utilizamos la función **matplotlib.plot()** para graficar, esta función viene desde la biblioteca matplotlib por lo tanto no fue un problema, se utilizaron los vectores ya hechos x - y como los datos de la gráfica. Después se guardo la gráfica en un PDF para posteriormente generar una secuencia.

I. Ejercicio número 9

En python con el mismo objetivo usamos uso de la libreria de pandas. Por lo cual las funciones correspondientes fueron **pandas.read_csv() y pandas.read_table()**

J. Ejercicio número 10

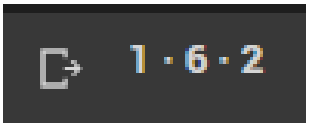
Nuestra forma de resolver esta practica fue de igual forma usando la librería de matplotlib. La cual usamos plot y boxplot que tiene una forma diferente de representar el diagrama, ademas implementamos histogramas, que matplotlib es bastante sencillo transformar de dispersion a histogramas

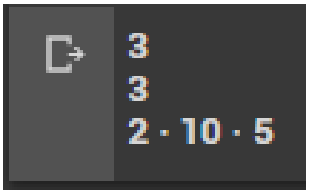
IV. Resultados

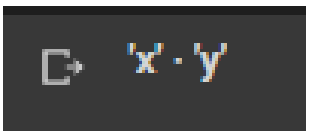
Es complicado mostrar estos de forma digerible, pero a continuación se muestran en el orden tal y como lo arroja nuestro programa. En el caso de R, se muestran todos los resultados de R que especifica el libro y en el mismo orden. En el de Python también están en el orden que especifica el libro, pero hay algunos que no se pueden realizar igual debido a que utilizan funciones especificas de R, por lo mismo la cantidad de resultados no es la misma.

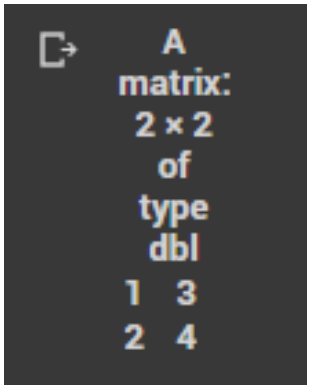
A. Ejercicios en R:

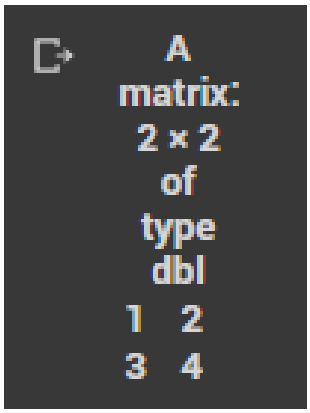
1) Actividad 1- Los resultados obtenidos para la Actividad 1 fueron los siguientes :

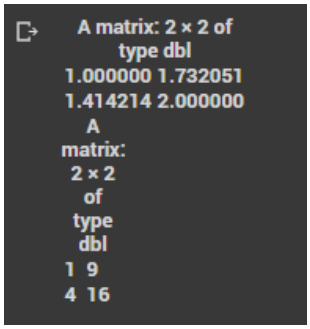


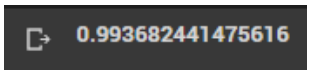


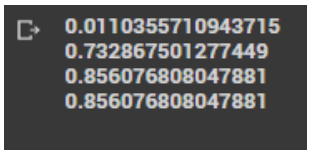




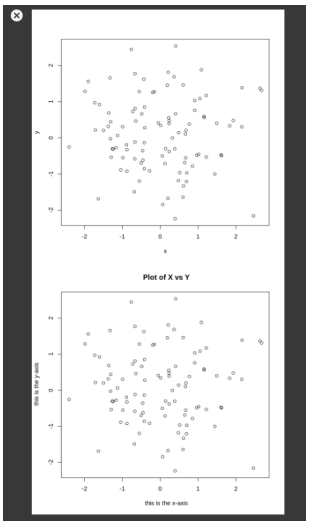


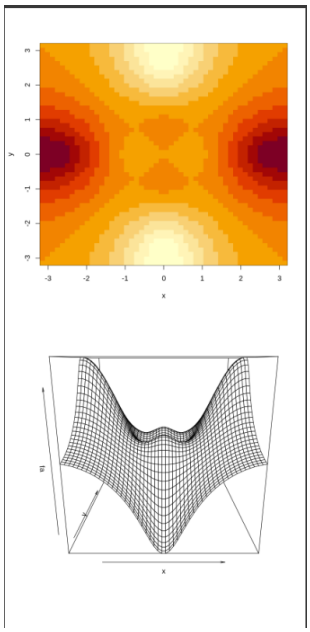






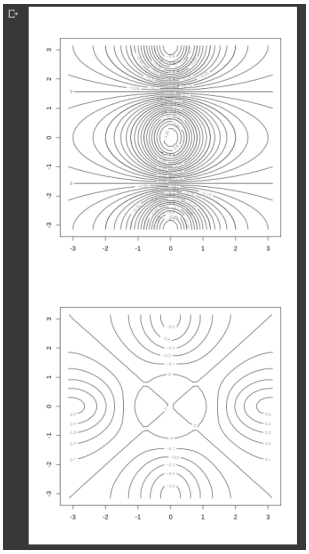
2) Actividad 2 - Los resultados fueron los siguientes :

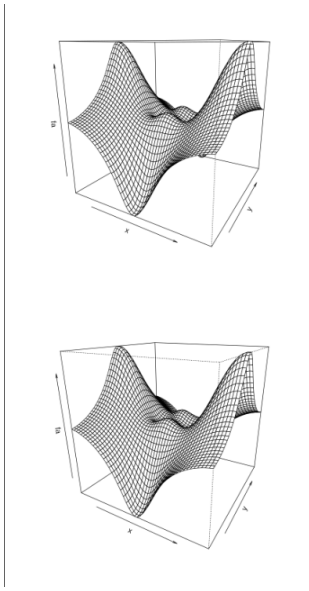


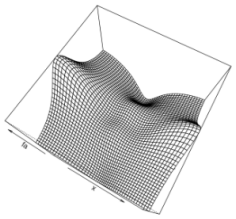


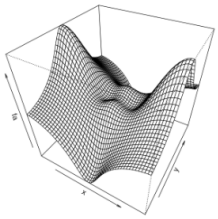
png: 2

1 · 2 · 3 · 4 · 5 · 6 · 7 · 8 · 9 · 10
1 · 2 · 3 · 4 · 5 · 6 · 7 · 8 · 9 · 10









3) Actividad 3 - Los resultados obtenidos para la actividad 3 son los siguientes

```
A matrix:
4 × 4 of
type int
1 5 9  13
2 6 10 14
3 7 11 15
4 8 12 16
```

```
A
matrix:
2 × 2
of
type
int
5  13
7  15
A
matrix:
3 × 3 of
type int
5 9  13
6 10 14
7 11 15
A matrix:
2 × 4 of
type int
1 5 9  13
2 6 10 14
A
matrix:
4 × 2
of
type
int
1  5
2  6
3  7
4  8
```

```
1 · 5 · 9 · 13
```

```
10
```

```
A matrix:
2 × 4 of
type int
2 6 10 14
4 8 12 16
6 · 8
```

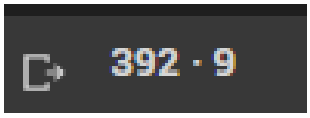
4) Actividad 4 - Los resultados obtenidos para la actividad 4 son los siguientes :

A data.frame: 398 × 9

V1	V2	V3	V4	V5	V6	V7	V8	V9
<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>	<fct>
mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
18.0	8	307.0	130.0	3504.	12.0	70	1	chevrolet chevelle malibu
15.0	8	350.0	165.0	3693.	11.5	70	1	buick skylark 320
18.0	8	318.0	150.0	3436.	11.0	70	1	plymouth satellite
16.0	8	304.0	150.0	3433.	12.0	70	1	amc rebel sst
17.0	8	302.0	140.0	3449.	10.5	70	1	ford torino
15.0	8	429.0	198.0	4341.	10.0	70	1	ford galaxie 500
14.0	8	454.0	220.0	4354.	9.0	70	1	chevrolet impala
14.0	8	440.0	215.0	4312.	8.5	70	1	plymouth fury iii
14.0	8	455.0	225.0	4425.	10.0	70	1	pontiac catalina
15.0	8	390.0	190.0	3850.	8.5	70	1	amc ambassador dpl
15.0	8	383.0	170.0	3563.	10.0	70	1	dodge challenger se
14.0	8	340.0	160.0	3609.	8.0	70	1	plymouth 'cuda 340
15.0	8	400.0	150.0	3761.	9.5	70	1	chevrolet monte carlo
14.0	8	455.0	225.0	3086.	10.0	70	1	buick estate wagon (sw)
24.0	4	113.0	95.00	2372.	15.0	70	3	toyota corona mark ii
22.0	6	198.0	95.00	2833.	15.5	70	1	plymouth duster
18.0	6	199.0	97.00	2774.	15.5	70	1	amc hornet
21.0	6	200.0	85.00	2587.	16.0	70	1	ford maverick
27.0	4	97.00	88.00	2130.	14.5	70	3	datsun pl510
26.0	4	97.00	46.00	1835.	20.5	70	2	volkswagen 1131 deluxe sedan
25.0	4	110.0	87.00	2672.	17.5	70	2	peugeot 504
24.0	4	107.0	90.00	2430.	14.5	70	2	audi 100 ls
25.0	4	104.0	95.00	2375.	17.5	70	2	saab 99e
26.0	4	121.0	113.0	2234.	12.5	70	2	bmw 2002
21.0	6	199.0	90.00	2648.	15.0	70	1	amc gremlin
10.0	8	360.0	215.0	4615.	14.0	70	1	ford f250
10.0	8	307.0	200.0	4376.	15.0	70	1	chevy c20
11.0	8	318.0	210.0	4382.	13.5	70	1	dodge d200
9.0	8	304.0	193.0	4732.	18.5	70	1	hi 1200d
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
28.0	4	112.0	88.00	2605.	19.6	82	1	chevrolet cavalier
27.0	4	112.0	88.00	2640.	18.6	82	1	chevrolet cavalier wagon
34.0	4	112.0	88.00	2395.	18.0	82	1	chevrolet cavalier 2-door
31.0	4	112.0	85.00	2575.	16.2	82	1	pontiac j2000 se hatchback
29.0	4	135.0	84.00	2525.	16.0	82	1	dodge aries se
27.0	4	151.0	90.00	2735.	18.0	82	1	pontiac phoenix
24.0	4	140.0	92.00	2865.	16.4	82	1	ford fairmont futura
36.0	4	105.0	74.00	1980.	15.3	82	2	volkswagen rabbit l
37.0	4	91.00	68.00	2025.	18.2	82	3	mazda glc custom l
31.0	4	91.00	68.00	1970.	17.6	82	3	mazda glc custom
38.0	4	105.0	63.00	2125.	14.7	82	1	plymouth horizon miser
36.0	4	98.00	70.00	2125.	17.3	82	1	mercury lynx l
36.0	4	120.0	88.00	2160.	14.5	82	3	nissan stanza xe
36.0	4	107.0	75.00	2205.	14.5	82	3	honda accord
34.0	4	108.0	70.00	2245	16.9	82	3	toyota corolla
38.0	4	91.00	67.00	1965.	15.0	82	3	honda civic
32.0	4	91.00	67.00	1965.	15.7	82	3	honda civic (auto)
38.0	4	91.00	67.00	1995.	16.2	82	3	datsun 310 gx
25.0	6	181.0	110.0	2945.	16.4	82	1	buick century limited
38.0	6	262.0	85.00	3015.	17.0	82	1	oldsmobile cutlass ciera (diesel)
26.0	4	156.0	92.00	2585.	14.5	82	1	chrysler lebaron medallion
22.0	6	232.0	112.0	2835	14.7	82	1	ford granada l
32.0	4	144.0	96.00	2665.	13.9	82	3	toyota celica gt

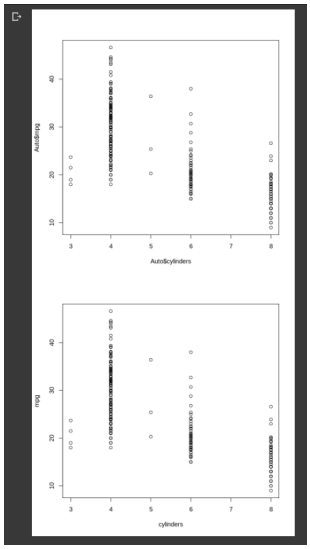
A data.frame: 397 × 9

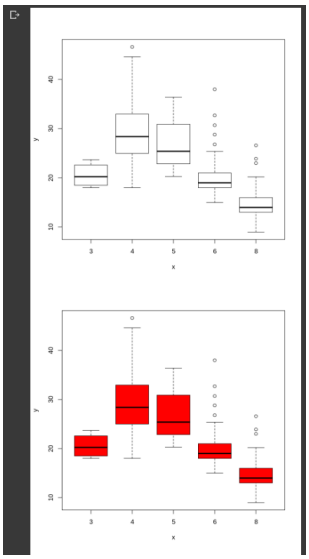
mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
<dbl>	<int>	<dbl>	<dbl>	<int>	<dbl>	<int>	<int>	<fct>
18	8	307	130	3504	12.0	70	1	chevrolet chevelle malibu
15	8	350	165	3693	11.5	70	1	buick skylark 320
18	8	318	150	3436	11.0	70	1	plymouth satellite
16	8	304	150	3433	12.0	70	1	amc rebel sst
17	8	302	140	3449	10.5	70	1	ford torino
15	8	429	198	4341	10.0	70	1	ford galaxie 500
14	8	454	220	4354	9.0	70	1	chevrolet impala
14	8	440	215	4312	8.5	70	1	plymouth fury iii
14	8	455	225	4425	10.0	70	1	pontiac catalina
15	8	390	190	3850	8.5	70	1	amc ambassador dpl
15	8	383	170	3563	10.0	70	1	dodge challenger se
14	8	340	160	3609	8.0	70	1	plymouth 'cuda 340
15	8	400	150	3761	9.5	70	1	chevrolet monte carlo
14	8	455	225	3086	10.0	70	1	buick estate wagon (sw)
24	4	113	95	2372	15.0	70	3	toyota corona mark ii
22	6	198	95	2833	15.5	70	1	plymouth duster
18	6	199	97	2774	15.5	70	1	amc hornet
21	6	200	85	2587	16.0	70	1	ford maverick
27	4	97	88	2130	14.5	70	3	datsun pl510
26	4	97	46	1835	20.5	70	2	volkswagen 1131 deluxe sedan
25	4	110	87	2672	17.5	70	2	peugeot 504
24	4	107	90	2430	14.5	70	2	audi 100 ls
25	4	104	95	2375	17.5	70	2	saab 99e
26	4	121	113	2234	12.5	70	2	bmw 2002
21	6	199	90	2648	15.0	70	1	amc gremlin
10	8	360	215	4615	14.0	70	1	ford f250
10	8	307	200	4376	15.0	70	1	chevy c20
11	8	318	210	4382	13.5	70	1	dodge d200
9	8	304	193	4732	18.5	70	1	hi 1200d
27	4	97	88	2130	14.5	71	3	datsun pl510
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
28	4	112	88	2605	19.6	82	1	chevrolet cavalier
27	4	112	88	2640	18.6	82	1	chevrolet cavalier wagon
34	4	112	88	2395	18.0	82	1	chevrolet cavalier 2-door
31	4	112	85	2575	16.2	82	1	pontiac j2000 se hatchback
29	4	135	84	2525	16.0	82	1	dodge aries se
27	4	151	90	2735	18.0	82	1	pontiac phoenix
24	4	140	92	2865	16.4	82	1	ford fairmont futura
36	4	105	74	1980	15.3	82	2	volkswagen rabbit l
37	4	91	68	2025	18.2	82	3	mazda glc custom l
31	4	91	68	1970	17.6	82	3	mazda glc custom
38	4	105	63	2125	14.7	82	1	plymouth horizon miser
36	4	98	70	2125	17.3	82	1	mercury lynx l
36	4	120	88	2160	14.5	82	3	nissan stanza xe
36	4	107	75	2205	14.5	82	3	honda accord
34	4	108	70	2245	16.9	82	3	toyota corolla
38	4	91	67	1965	15.0	82	3	honda civic
32	4	91	67	1965	15.7	82	3	honda civic (auto)
38	4	91	67	1995	16.2	82	3	datsun 310 gx
25	6	181	110	2945	16.4	82	1	buick century limited
38	6	262	85	3015	17.0	82	1	oldsmobile cutlass ciera (diesel)
26	4	156	92	2585	14.5	82	1	chrysler lebaron medallion
22	6	232	112	2835	14.7	82	1	ford granada l
32	4	144	96	2665	13.9	82	3	toyota celica gt

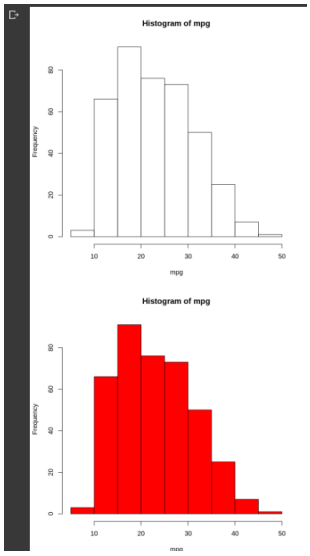


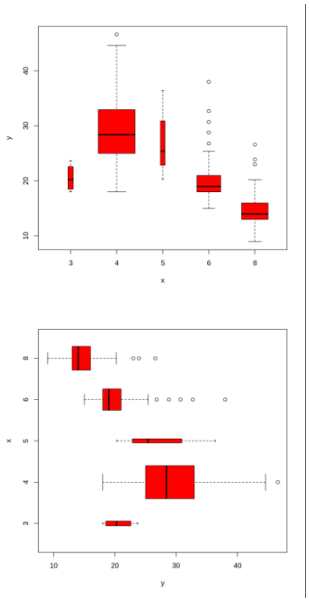
'mpg' · 'cylinders' · 'displacement' · 'horsepower' · 'weight' · 'acceleration' · 'year' · 'origin' · 'name'

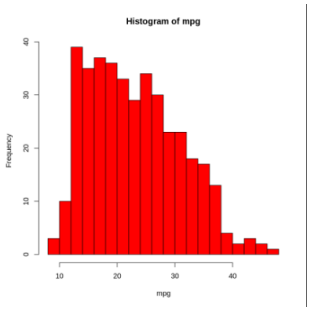
5) Actividad 5 - Los resultados para la actividad 5 son los siguientes :

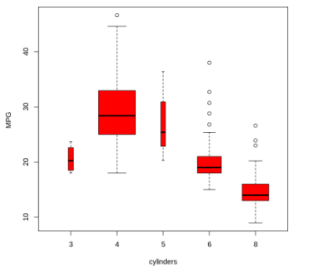


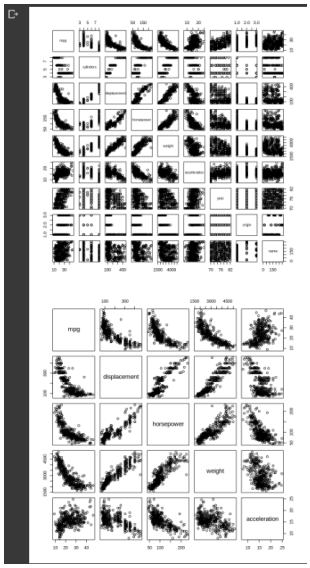

y
x
3
4
5
6
8
10
20
30
40

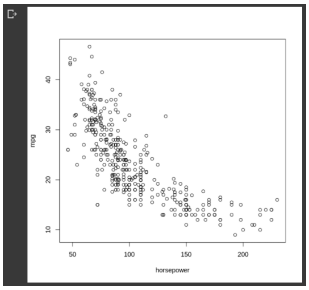

Histogram of mpg
Frequency
mpg
0
20
40
60
80
10
20
30
40
50

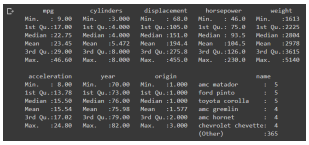

y
x
3
4
5
6
8
10
20
30
40


Histogram of mpg
Frequency
mpg
0
10
20
30
40
10
20
30
40


MPG
cylinders
3
4
5
6
8
10
20
30
40

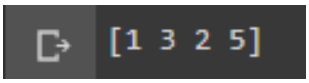

mpg
displacement
horsepower
weight
acceleration

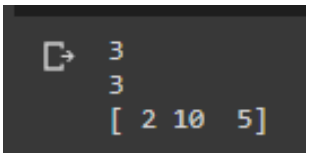




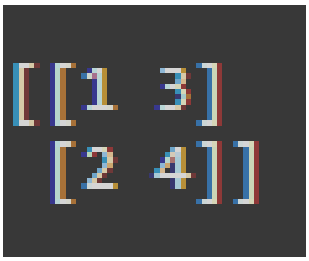
B. Ejercicios en Python:

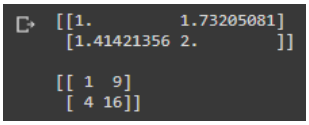
6) Actividad 6 - Los resultados de la actividad 6 son las siguientes:

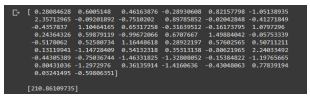


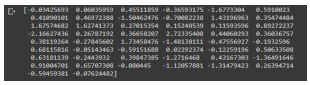


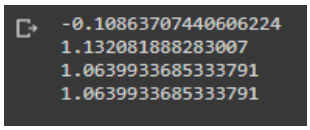
Aquí deberían ir dos imágenes de resultados que no caben, pero muestran los elementos en el directorio, incluyendo todas las variables creadas por usuario y por el sistema. La primera es antes de borrar los elementos creados por el usuario y la segunda es después de borrarlos.



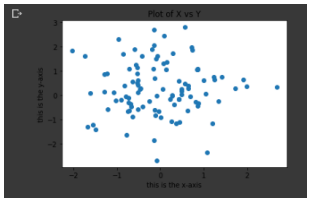


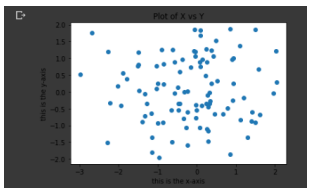


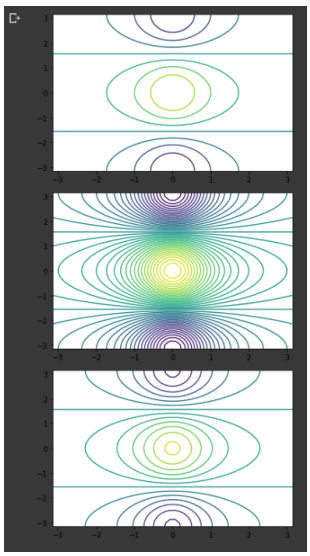


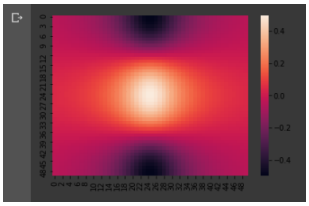


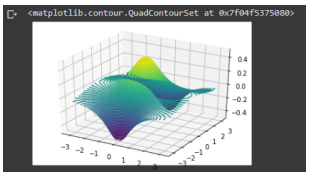
7) Actividad 7 - Los resultados de la actividad 7 son los siguientes:











8) Actividad 8 - Los resultados de la actividad 8 son los siguientes:

```
[[ 1  5  9 13]
 [ 2  6 10 14]
 [ 3  7 11 15]
 [ 4  8 12 16]]
```

```
10
```

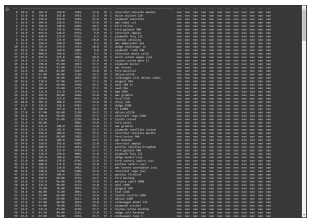
```
[[ 5 13]
 [ 7 15]]

[[ 5  9 13]
 [ 6 10 14]
 [ 7 11 15]]

[[ 1  5  9 13]
 [ 2  6 10 14]]

[[1 5]
 [2 6]
 [3 7]
 [4 8]]
```

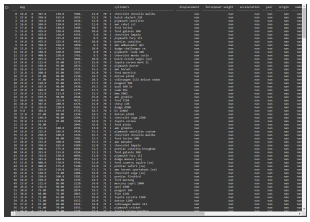
```
[[ 1  5  9 13]]
```

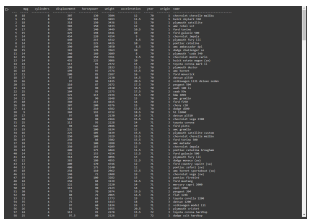
```
[[ 2  6 10 14]
 [ 4  8 12 16]]

[[6]
 [8]]
```

```
(4, 4)
```

9) Actividad 9 - Los resultados de la actividad 9 son los siguientes:



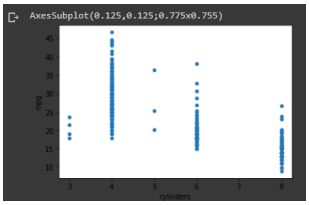


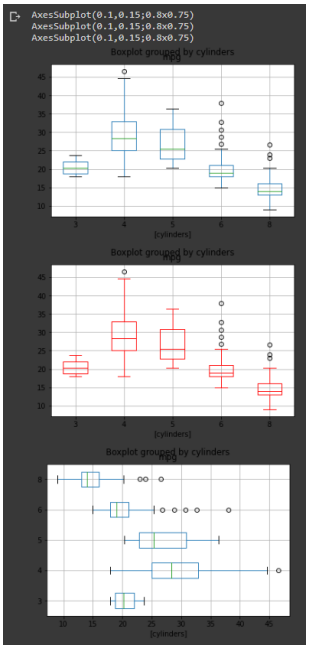


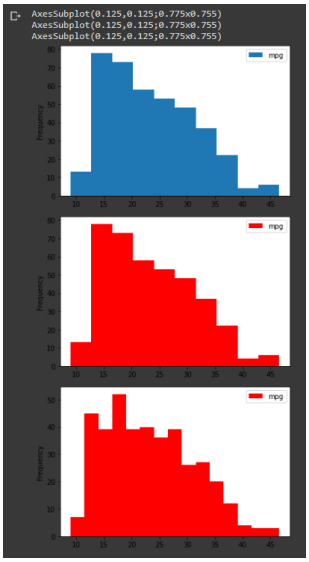
```
(392, 9)
```

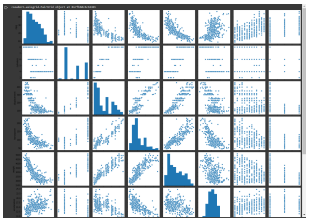
```
mpg
cylinders
displacement
horsepower
weight
acceleration
year
origin
name
```

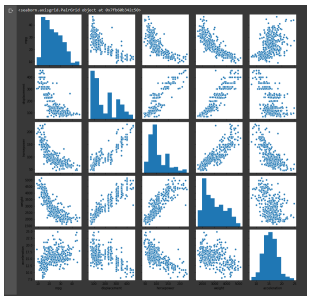
10) Actividad 10 - Los resultados de la actividad 10 son los siguientes:

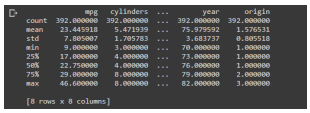






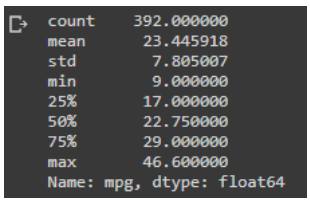






	mpg	cylinders	...	year	origin
count	392.000000	392.000000	...	392.000000	392.000000
mean	23.445918	5.471939	...	75.979592	1.576531
std	7.805007	1.705783	...	3.683737	0.805518
min	9.000000	3.000000	...	70.000000	1.000000
25%	17.000000	4.000000	...	73.000000	1.000000
50%	22.750000	4.000000	...	76.000000	1.000000
75%	29.000000	8.000000	...	79.000000	2.000000
max	46.600000	8.000000	...	82.000000	3.000000

[8 rows x 8 columns]



```
count    392.000000
mean      23.445918
std        7.805007
min        9.000000
25%       17.000000
50%       22.750000
75%       29.000000
max       46.600000
Name: mpg, dtype: float64
```

V. CONCLUSIONES

En esta práctica se definieron algunas funciones tanto de R como de python además de que se demostró un ejemplo de cada una de estas. Esta fue una buena práctica un poco más compleja que la anterior, pero fue posible solucionar los problemas. Al principio pensamos que iban a ser unos códigos intraducibles para python ya que el código principal estaba diseñado en r y que iban a ser muy complejos pero realmente no terminó siendo tan complicado. Python maneja muchas librerías para data science como por ejemplo numpy, además para graficar esta matplotlib. Para concluir, podemos decir que fue una práctica sencilla dentro del acondicionamiento de la materia, y esperamos seguir reforzando los conocimientos de R.