

Notas Generales: Módulo 1

Laboratorios

Preguntas

Nombre: **Francisco Javier Ramírez Arias**

Matricula: **A01316379**

Objetivos

- Adquisición de Datos
- ¿Cómo obtener información básica de los datos?
- Data Wrangling
- Análisis de Datos Exploratorio
- Desarrollo de modelos
- Evaluación de **modelos**

▼ Introducción de al Análisis de Datos con Python

- Los problemas requieren del análisis de datos.
- Un conjunto de datos analizado con Python
- Introducción a los paquetes de análisis de Python
- Importar y Exportar datos in Python
- Información Básica del conjunto de datos

Problema a analizar, estimación del precio de los autos usados, por medio de la limpieza de los datos, analisis exploratorio, desarrollo de modelos y evaluacion de modelos.

¿Por qué el análisis de Datos?

- Los datos estan en todas partes. El análisis de datos/ciencia de datos ayuda a contestar preguntas de los datos
- El análisis de datos juega un importante rol en:
 1. Descubri informacion util.
 2. Responder preguntas.
 3. Predecir el futuro o lo que no se sabe.

Entendiendo los datos.

- Conjunto de datos de autos usados.

- Se encuentra en formato CSV.
- Cada atributo en el conjunto de datos.
- Predecir el precio de autos usados, problema (regresion).

Paquetes de Python para Ciencia de Datos

Librerías de Computo Científico

- Pandas: (Estructura de datos, herramientas)
- Numpy: (Arreglos y matrices)
- SciPy: (Integrales, Ecuaciones Diferenciales, Optimización)

Librerías de Visualización

- Matplotlib (gráficas)
- Seaborn (mapas de calor, series de tiempo, etc)

Librería de Algoritmos

- Scikit-learn (Machine learning: regresión, clasificación,...)
- Statsmodels (Explora datos, estima modelos y realiza pruebas estadísticas)

Importando y Exportando Datos en Python

Importando Datos

- Proceso de cargar y leer datos dentro de Python de diferentes fuentes.
- Tiene dos propiedades importantes:
 1. Formato (.csv, .json, .xlsx, .hdf)
 2. Ruta del archivo de datos (Computer:/Desktop/mydata.csv)

Obteniendo los datos

```
import pandas as pd

url = "/content/my_data/imports-85.data"

df = pd.read_csv(url, header = None)

#Imprime todo el conjunto
df
#Imprime Los primeros renglones
#df.head(n)
#Imprime los ultimos renglones
#df.tail()
```

```
#Agregar encabezado, reemplazando el encabezado por defecto.
headers =["symboling","normalized-losses","make","fuel-type","aspiration","num-of-doors","body-style",
          "wheel-base","length","width","height","curb-weight","engine-type","num-of-cylinders","engine
          "compression-ratio","horsepower","peak-rpm","city-mpg","highway-mpg","price"]
df.columns = headers
df.head()
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	e
0	3	?	alfa-romero	gas	std	two	convertible	rwd	
1	3	?	alfa-romero	gas	std	two	convertible	rwd	
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	
3	2	164	audi	gas	std	four	sedan	fwd	
4	2	164	audi	gas	std	four	sedan	4wd	

5 rows × 26 columns



Exportando los datos

```
path= "/content/my_data/automobile.csv"
df.to_csv(path)
```

Exportando a diferentes formatos en Python

cvs pd.read_csv() df.to_csv()

json pd.read_json() df.to_json()

Excel pd.read_excel() df.to_excel()

sql pd.read_sql() df.to_sql()

Empezando analizar Datos en Python

- Entender los datos antes de comenzar a analizarlos
- Tipos de datos
- Distribución de los datos

Tipos de datos

En Pandas

- Objetos (numero y cadenas de caracteres)
- int64 (caracteres numericos)
- float64 (caracteres numericos con decimales)
- datetime64, timedelta[ns] (datos de tiempo)

Nativo de Python

- String (numero y cadenas de caracteres)
- int (caracteres numericos)
- float (caracteres numericos con decimales)
- N/A (datos de tiempo)

Por que revisar los datos? Informacion potencial y por el tipo de datos a manejar. Compatibilidad con los metodos de Python.

- Utilizamos dataframe.dtypes "df.dtypes", para revisar el tipo de datos.
- Utilizamos dataframe.describe "df.describe()", para resúmenes estadísticos.
- Utilizamos dataframe.describe(include="all"), para un resumen completo de estadísticas.
- Utilizamos dataframe.info(). para un resumen conciso de los datos.

▼ Laboratorio 1

Adquisicion de Datos

```
#Cargamos el conjunto de datos, este se encuentra previamente descargado en el entorno de COLab
import pandas as pd
path= "/content/my_data/imports-85.data"
df = pd.read_csv(path, header=None)
```

```
#Imprime los primeros 5 renglones
print("The first 5 rows of the dataframe")
df.head(5)
```

The first 5 rows of the dataframe

	0	1	2	3	4	5	6	7	8	9	...	16	17	18
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68
3	2	164	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19

▼ Pregunta: 1

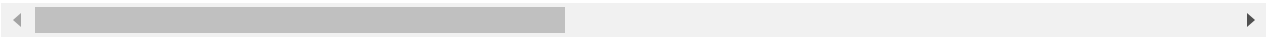
Revisa los 10 ultimos renglones del dataframe

```
print("Los 10 ultimos renglones del dataframe")
df.tail(10)
```

Los 10 ultimos renglones del dataframe

	symboling	normalized- losses	make	fuel- type	aspiration	num- of- doors	body- style	drive- wheels	engi locat
195	-1	74	volvo	gas	std	four	wagon	rwd	f
196	-2	103	volvo	gas	std	four	sedan	rwd	f
197	-1	74	volvo	gas	std	four	wagon	rwd	f
198	-2	103	volvo	gas	turbo	four	sedan	rwd	f
199	-1	74	volvo	gas	turbo	four	wagon	rwd	f
200	-1	95	volvo	gas	std	four	sedan	rwd	f
201	-1	95	volvo	gas	turbo	four	sedan	rwd	f
202	-1	95	volvo	gas	std	four	sedan	rwd	f
203	-1	95	volvo	diesel	turbo	four	sedan	rwd	f
204	-1	95	volvo	gas	turbo	four	sedan	rwd	f

10 rows × 26 columns



Agregamos encabezados al archivo

```
#Se crea la lista del encabezado
headers = ["symboling","normalized-losses","make","fuel-type","aspiration", "num-of-doors","body-style",
           "drive-wheels","engine-location","wheel-base", "length","width","height","curb-weight","engine-
           num-of-cylinders", "engine-size","fuel-system","bore","stroke","compression-ratio","horsepower",
           "peak-rpm","city-mpg","highway-mpg","price"]
print("headers\n", headers)
```

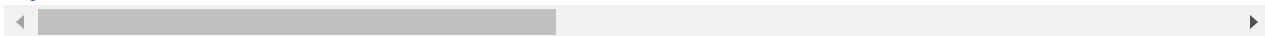
```
headers
['symboling', 'normalized-losses', 'make', 'fuel-type', 'aspiration', 'num-of-doors', 'body-styl
```



```
#Reemplazamos el encabezado y revisamos el dataframe
df.columns = headers
df.head(10)
```

	symboling	normalized- losses	make	fuel- type	aspiration	num- of- doors	body- style	drive- wheels	1
0	3	NaN	alfa-romero	gas	std	two	convertible	rwd	
1	3	NaN	alfa-romero	gas	std	two	convertible	rwd	
2	1	NaN	alfa-romero	gas	std	two	hatchback	rwd	
3	2	164	audi	gas	std	four	sedan	fwd	
4	2	164	audi	gas	std	four	sedan	4wd	
5	2	NaN	audi	gas	std	two	sedan	fwd	
6	1	158	audi	gas	std	four	sedan	fwd	
7	1	NaN	audi	gas	std	four	wagon	fwd	
8	1	158	audi	gas	turbo	four	sedan	fwd	
10	2	192	bmw	gas	std	two	sedan	rwd	

10 rows × 26 columns



#Reemplazamos el simbolo de interrogación con NaN

```
df1=df.replace('?',np.NaN)
```

#Eliminamos los valores faltantes de la columna de precio

```
df=df1.dropna(subset=["price"], axis=0)
```

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

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels
0	3	NaN	alfa-romero	gas	std	two	convertible	rwd
1	3	NaN	alfa-romero	gas	std	two	convertible	rwd
2	1	NaN	alfa-romero	gas	std	two	hatchback	rwd
3	2	164	audi	gas	std	four	sedan	fwd
4	2	164	audi	gas	std	four	sedan	4wd
5	2	NaN	audi	gas	std	two	sedan	fwd
6	1	158	audi	gas	std	four	sedan	fwd
7	1	NaN	audi	gas	std	four	wagon	fwd
8	1	158	audi	gas	turbo	four	sedan	fwd
10	2	192	bmw	gas	std	two	sedan	rwd
11	0	192	bmw	gas	std	four	sedan	rwd

▼ Pregunta: 2

Encuentra el nombre de las columnas del dataframe

```

15         0         NaN         BMW         gas         std         four         sedan         rwd

```

```

#Imprime en una lista el nombre de las columnas
print(df.columns)

Index(['symboling', 'normalized-losses', 'make', 'fuel-type', 'aspiration',
      'num-of-doors', 'body-style', 'drive-wheels', 'engine-location',
      'wheel-base', 'length', 'width', 'height', 'curb-weight', 'engine-type',
      'num-of-cylinders', 'engine-size', 'fuel-system', 'bore', 'stroke',
      'compression-ratio', 'horsepower', 'peak-rpm', 'city-mpg',
      'highway-mpg', 'price'],
      dtype='object')

```

```

#Guardamos los datos
df.to_csv("/content/my_data/autos.csv", index=False)

```

```

#Nos muestra el tipo de dato de cada columna
df.dtypes

```

```

symboling          int64
normalized-losses  object
make              object
fuel-type          object
aspiration         object
num-of-doors       object
body-style         object

```

```
drive-wheels      object
engine-location   object
wheel-base       float64
length           float64
width            float64
height           float64
curb-weight       int64
engine-type       object
num-of-cylinders  object
engine-size       int64
fuel-system       object
bore             object
stroke           object
compression-ratio float64
horsepower        object
peak-rpm          object
city-mpg          int64
highway-mpg       int64
price            object
dtype: object
```

```
#Imprime el tipo de datos de cada columna
print(df.dtypes)
```

```
symboling         int64
normalized-losses object
make             object
fuel-type         object
aspiration        object
num-of-doors      object
body-style        object
drive-wheels      object
engine-location   object
wheel-base       float64
length           float64
width            float64
height           float64
curb-weight       int64
engine-type       object
num-of-cylinders  object
engine-size       int64
fuel-system       object
bore             object
stroke           object
compression-ratio float64
horsepower        object
peak-rpm          object
city-mpg          int64
highway-mpg       int64
price            object
dtype: object
```

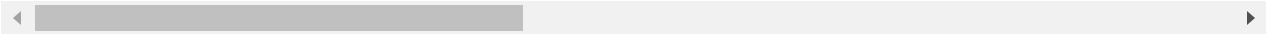
```
#Resumen estadístico
df.describe()
```


	symboling	wheel- base	length	width	height	curb- weight	e
count	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.0
mean	0.840796	98.797015	174.200995	65.889055	53.766667	2555.666667	126.0
std	1.254802	6.066366	12.322175	2.101471	2.447822	517.296727	41.0
min	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.0
25%	0.000000	94.500000	166.800000	64.100000	52.000000	2169.000000	98.0
50%	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.0


```
#Resumen estadístico
df.describe(include="all")
```

	symboling	normalized- losses	make	fuel- type	aspiration	num- of- doors	body- style	drive- wheels	lo
count	201.000000	164	201	201	201	199	201	201	
unique	NaN	51	22	2	2	2	5	3	
top	NaN	161	toyota	gas	std	four	sedan	fwd	
freq	NaN	11	32	181	165	113	94	118	
mean	0.840796	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
std	1.254802	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
min	-2.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
25%	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
50%	1.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
75%	2.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
max	3.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

11 rows × 26 columns



```
#Se aplica el metodo .describe() a dos columnas
df[['length','compression-ratio']].describe()
```

	length	compression-ratio	
count	201.000000	201.000000	
mean	174.200995	10.164279	
std	12.322175	4.004965	
min	141.100000	7.000000	

#Obtener informacion del dataframe

df.info()

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 201 entries, 0 to 204
Data columns (total 26 columns):
#   Column                Non-Null Count  Dtype
---  -
0   symboling              201 non-null    int64
1   normalized-losses      164 non-null    object
2   make                   201 non-null    object
3   fuel-type              201 non-null    object
4   aspiration              201 non-null    object
5   num-of-doors           199 non-null    object
6   body-style             201 non-null    object
7   drive-wheels           201 non-null    object
8   engine-location        201 non-null    object
9   wheel-base            201 non-null    float64
10  length                 201 non-null    float64
11  width                  201 non-null    float64
12  height                 201 non-null    float64
13  curb-weight            201 non-null    int64
14  engine-type            201 non-null    object
15  num-of-cylinders       201 non-null    object
16  engine-size            201 non-null    int64
17  fuel-system            201 non-null    object
18  bore                   197 non-null    object
19  stroke                 197 non-null    object
20  compression-ratio      201 non-null    float64
21  horsepower             199 non-null    object
22  peak-rpm               199 non-null    object
23  city-mpg               201 non-null    int64
24  highway-mpg            201 non-null    int64
25  price                  201 non-null    object
dtypes: float64(5), int64(5), object(16)
memory usage: 42.4+ KB
```

Final del módulo

Preguntas (Mini-Cuestionario)

1. What does CSV stand for? **Comma-separated values**
2. In the data set, which of the following represents an attribute or feature? **Column**
3. What is the name of what we want to predict? **Target**
4. What is the command to display the first five rows of a dataframe df? **df.head()**

5. What command do you use to get the data type of each row of the dataframe df? **df.dtypes**
6. How do you get a statistical summary of a dataframe df? **df.describe()**
7. If you use the method describe() without changing any of the arguments, you will get a statistical summary of all the columns of type "object". **False**

[Productos de pago de Colab](#) - [Cancelar contratos](#)

✓ 0 s completado a las 20:04



Course Progress for 'Francisco_Arias' (A01316379@tec.mx)

