

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. Some nodes are highlighted with blue circles, and others with blue dots. The diagram is composed of light gray lines and dots, with a few blue accents.

# Predicciones del Rendimiento del Combustible de un Auto

Alejandra Silva H.  
06-10-2022

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, featuring a complex web of interconnected nodes and lines. Some nodes are highlighted with blue circles, and others with blue dots. The diagram is composed of light gray lines and dots, with a few blue accents.

# Contenidos

- Introducción
- Concepto Clave
- Conjunto de Datos
- Revisión
- Limpieza
- Visualizaciones
- Desafío



1.

# Introducción

Hoy en día se considera fundamental cuidar el medioambiente debido al cambio climático y hay pocos contaminantes tan severos como el petróleo.

¿Se puede hacer algo para minimizar la huella de carbono? Lo primero que se debe hacer es tomar conciencia de este problema e informarse.

En el escenario actual, en que las ventas de autos han superado los récords de años anteriores, ¿se puede tomar alguna decisión que impacte menos en el medioambiente?

¿Se puede predecir el rendimiento del combustible de un auto?



# 2.

## Concepto Clave

### Rendimiento


Es la relación que hay entre la distancia que un auto puede recorrer y la cantidad de litros de combustible consumidos para recorrer dicha distancia; se expresa en millas por galón (mpg) o en kilómetros por litro (km/l).

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines, with some nodes highlighted in blue.

# 3.

## Conjunto de Datos

Este conjunto de datos tiene 398 filas y 9 columnas y entrega millaje, caballos de fuerza, año del modelo y otras especificaciones de automóviles.



| Column Position | Attribute Name | Description   | Examples           |
|-----------------|----------------|---|--------------------|
| #1              | mpg            | fuel efficiency measured in miles per gallon (mpg)    | 9.0, 13.0, 41.5    |
| #2              | cylinders      | number of cylinders in the engine                     | 3, 4, 8            |
| #3              | displacement   | engine displacement (in cubic inches)                 | 68.0, 112.0, 455.0 |
| #4              | horsepower     | engine horsepower                                     | 46.0, 70.0, 230.0  |
| #5              | weight         | vehicle weight (in pounds)                            | 1613, 3615, 5140   |
| #6              | acceleration   | time to accelerate from 0 to 60 mph (in seconds)      | 8.00, 15.50, 24.80 |
| #7              | model year     | model year  | 73, 79, 82         |
| #8              | origin         | origin of car (1: American, 2: European, 3: Japanese) | 1, 2, 3            |
| #9              | car name       | car name  | audi fox, subaru   |

# 4.

## Revisión

- © Lo primero que se hizo fue revisar si el df se había cargado bien usando `dfAutos.head()`.
- © Se identificó que los nombres de las columnas no venían y se agregaron de la siguiente manera:  

```
dfAutos.columns = ['mpg', 'cylinders', 'displacement', 'horsepower', 'weight',  
'acceleration', 'model year', 'origin', 'brand']
```
- © Se validó si habían quedado bien con `dfAutos.head()`

|   | mpg  | cylinders | displacement | horsepower | weight | acceleration | model year | origin | brand |
|---|------|-----------|--------------|------------|--------|--------------|------------|--------|-------|
| 0 | 15.0 | 8         | 350.0        | 165        | 3693   | 11.5         | 70         | 1      | buick |

# 5.

## Limpieza

- © Con `dfAutos.info()` vemos toda la información relacionada a la cantidad de filas y columnas, tipos de dato, nombres de columnas y si hay datos nulos
- © Se ve que hay dos columnas de tipo `object` y se revisan los datos que contienen para ver si corresponde que sean de tipo texto o no
- © Horsepower debe ser de tipo numérico
- © Brand es de tipo texto

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 397 entries, 0 to 396  
Data columns (total 9 columns):  
#   Column          Non-Null Count  Dtype    
--  --    
0   mpg             397 non-null   float64   
1   cylinders       397 non-null   int64     
2   displacement    397 non-null   float64   
3   horsepower      397 non-null   object    
4   weight         397 non-null   int64     
5   acceleration    397 non-null   float64   
6   model year     397 non-null   int64     
7   origin         397 non-null   int64     
8   brand          397 non-null   object    
dtypes: float64(3), int64(4), object(2)  
memory usage: 28.0+ KB
```

|   | mpg  | cylinders | displacement | horsepower | weight | acceleration | model year | origin | brand    |
|---|------|-----------|--------------|------------|--------|--------------|------------|--------|----------|
| 0 | 15.0 | 8         | 350.0        | 165        | 3693   | 11.5         | 70         | 1      | buick    |
| 1 | 18.0 | 8         | 318.0        | 150        | 3436   | 11.0         | 70         | 1      | plymouth |
| 2 | 16.0 | 8         | 304.0        | 150        | 3433   | 12.0         | 70         | 1      | amc      |
| 3 | 17.0 | 8         | 302.0        | 140        | 3449   | 10.5         | 70         | 1      | ford     |
| 4 | 15.0 | 8         | 429.0        | 198        | 4341   | 10.0         | 70         | 1      | ford     |



Se revisaron los datos y se identificó que en la columna venía un caracter extraño. Era el caracter “?” Se hizo un replace por cero y después se convirtió el tipo de dato a int 64.

```
dfAutos['horsepower'] = dfAutos['horsepower'].replace('?', 0)
```

```
dfAutos['horsepower'] = dfAutos['horsepower'].astype('int64')
```

Se detectó 1 fila repetida. El 91% de los datos son iguales, siendo la única diferencia el año. La fila se elimina.

```
dfAutos = dfAutos[~((dfAutos['mpg'] == 27) & (dfAutos['cylinders'] == 4) & (dfAutos['displacement'] == 97) & (dfAutos['horsepower'] == 88) & (dfAutos['weight'] == 2130) & (dfAutos['acceleration'] == 14.5) & (dfAutos['model year'] == 71)))
```

En este dataset los datos iguales a cero se interpretan como falta de información por la naturaleza de los datos y considerar ceros en los datos sería un error que entregaría resultados basados en premisas erróneas. Ej. Un auto no puede tener 0 cilindro o una cilindrada de 0. Tampoco puede tener 0 caballos de fuerza ni tiempo de aceleración de 0 a 60 mph de 0 segundos.

Por lo anterior, se revisaron una a una las columnas en busca de datos en cero y los resultados fueron los siguientes:

```
datosCero = dfAutos[dfAutos['mpg'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['cylinders'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['displacement'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['horsepower'] == 0] # 6 *
```

```
datosCero = dfAutos[dfAutos['weight'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['acceleration'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['model year'] == 0] # NO HAY
```

```
datosCero = dfAutos[dfAutos['origin'] == 0] # NO HAY
```

\* Solo en la columna **horsepower** habían ceros por el replace del caracter extraño hecho anteriormente. Esos datos se **10** eliminaron para no afectar los resultados. Se eliminó el 1,51% de los datos, quedando para análisis el 98,49% (391 filas).

Todas las columnas con información de números quedaron con tipo de dato int o float y se eliminaron los datos cero, limpiando de esa manera el conjunto.

Se ve la eliminación de los registros con cero (de 397 bajó a 391) y que la columna **horsepower** cambió a tipo de dato int64.

La única columna que quedó como texto es la de los modelos de autos, como corresponde.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 391 entries, 0 to 396
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   mpg             391 non-null   float64
1   cylinders       391 non-null   int64
2   displacement    391 non-null   float64
3   horsepower      391 non-null   int64
4   weight          391 non-null   int64
5   acceleration    391 non-null   float64
6   model year      391 non-null   int64
7   origin          391 non-null   int64
8   brand           391 non-null   object
dtypes: float64(3), int64(5), object(1)
memory usage: 30.5+ KB
```

# 6.


## Visualizaciones

La visualización que fue de mayor utilidad fue, por lejos, `dfAutos.info()`. La información que se mostró fue fundamental para hacer la limpieza de datos requerida ya que indicó:

- ⦿ La cantidad de registros
- ⦿ La cantidad de columnas
- ⦿ Los nombres de las columnas
- ⦿ Si hay datos nulos
- ⦿ El tipo de dato de cada columna

Esta información permitió detectar errores en la data cuando una columna aparecía como `object` y encontrar si había datos nulos, lo que en esta oportunidad no ocurrió.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 391 entries, 0 to 396
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
---  -
0   mpg             391 non-null   float64
1   cylinders       391 non-null   int64
2   displacement    391 non-null   float64
3   horsepower      391 non-null   int64
4   weight          391 non-null   int64
5   acceleration    391 non-null   float64
6   model year      391 non-null   int64
7   origin          391 non-null   int64
8   brand           391 non-null   object
dtypes: float64(3), int64(5), object(1)
memory usage: 30.5+ KB
```



|   | mpg  | cylinders | displacement | horsepower | weight | acceleration | model year | origin | brand    |
|---|------|-----------|--------------|------------|--------|--------------|------------|--------|----------|
| 0 | 15.0 | 8         | 350.0        | 165        | 3693   | 11.5         | 70         | 1      | buick    |
| 1 | 18.0 | 8         | 318.0        | 150        | 3436   | 11.0         | 70         | 1      | plymouth |
| 2 | 16.0 | 8         | 304.0        | 150        | 3433   | 12.0         | 70         | 1      | amc      |
| 3 | 17.0 | 8         | 302.0        | 140        | 3449   | 10.5         | 70         | 1      | ford     |
| 4 | 15.0 | 8         | 429.0        | 198        | 4341   | 10.0         | 70         | 1      | ford     |

Otra visualización de ayuda fue `dfAutos.head()`, que permite ver los primeros 5 registros del dataset. Esto ayuda a ver si el dataset se cargó correctamente, facilita la revisión de los datos y si vienen los nombres de las columnas.



# 7.

## Desafío

Diría que el mayor desafío que presentó este conjunto de datos fue determinar si los datos 0 podían ser parte de la muestra y anticiparse al impacto que estos datos 0 podrían tener en los resultados.

Es fundamental considerar que cada conjunto de datos es un universo en sí mismo por la complejidad que puede llegar a tener y es tremendamente relevante no perder el foco del problema y tomar una decisión responsable respecto al manejo de los datos 0 o null.

Cualquier error de criterio al momento de limpiar los datos podría dejarlos totalmente inutilizables si no se revisan con el problema en mente. Es fácil caer en soluciones simples como completar los datos nulos con 0 o no fijarse en detalles por hacer este proceso a la rápida, lo que puede impactar enormemente en nuestro futuro modelo.



# Predicciones del Rendimiento del Combustible de un Auto

Alejandra Silva H.  
06-10-2022

