Análisis de economía de combustible

Este es un ejemplo de análisis de datos en datos históricos de economía de combustible. Tenemos datos de varios vehículos hechos desde el 2000 hasta el 2012. En este reporte, vamos a automatizar la importación de datos desde hojas de calculo, la creación de gráficas personalizadas, la partición de los datos en grupos, y la aplicación interactiva de ajustes de curvas específicos.

Podemos compartir nuestris resultados a través de una app interactiva o al exportar nuestro Live Script como un PDF para nuestro análisis.

Tabla de contenido

Importación de datos como tabla

Lo primero es traer los datos a MATLAB. La función importData fue creada automáticamente usando el Import Tool, de tal forma que se generó código que refleja nuestras acciones interactivas. Esto nos permite personalizar la importación sin pensar en programarlo todo nosotros mismos.

```
data = importData(2007);
```

```
head(data(:,["CarLine" "Car_Truck" "City_Highway" "Km_gal" "RatedHP" "Weight"]))
```

CarLine	Car_Truck	City_Highway	Km_gal	RatedHP	Weight
RAM 1500 PICKUP 2WD	truck	highway	41.199	345	1587.6
RAM 1500 PICKUP 2WD	truck	city	26.232	345	1587.6
LIBERTY/CHEROKEE 2WD	truck	highway	46.832	210	1814.4
LIBERTY/CHEROKEE 2WD	truck	city	30.256	210	1814.4
LIBERTY/CHEROKEE 4WD	truck	highway	45.062	210	1927.8
LIBERTY/CHEROKEE 4WD	truck	city	30.095	210	1927.8
PT CRUISER	truck	highway	54.878	220	1644.3
PT CRUISER	truck	city	37.015	220	1644.3

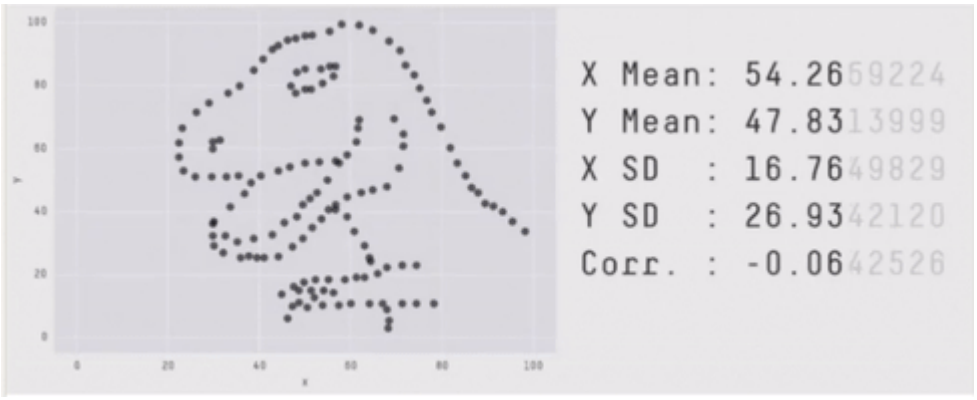
Usando la función summary podemos ver un resumen con algunas medidas estadísticas de las variables que deseamos analizar.

```
summary(data(:,{'RatedHP','Km_gal','Car_Truck'}))
```

```
Variables:

    RatedHP: 2595×1 double

        Values:

            Min          76
            Median      236
            Max         631

    Km_gal: 2595×1 double

        Values:

            Min        15.772
            Median     39.912
            Max        107.18

    Car_Truck: 2595×1 categorical

        Values:

            car        1343
            truck      1252
```

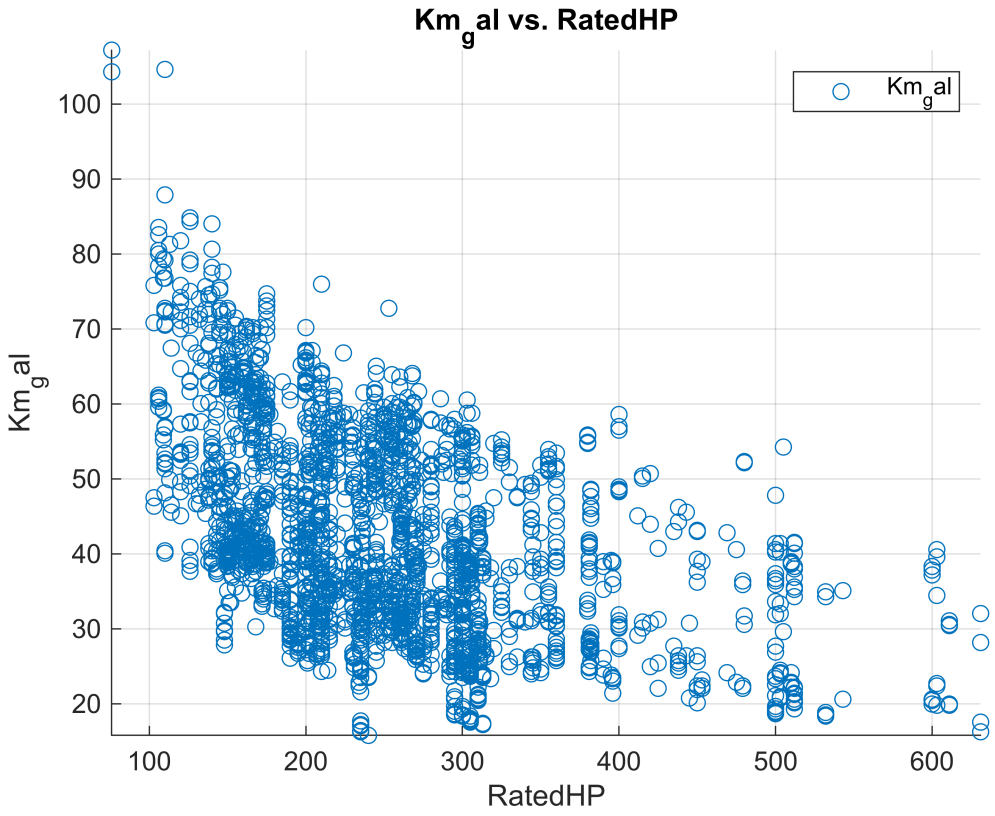
Visualización de los datos

Visualizar datos es muy necesario, por más seguros que estemos de sus medidas estadísticas. En la siguiente animación vemos cómo varios tipos de datos con dos decimales de precisión pueden tener un significado muy distinto.



A través de las tareas interactivas podemos crear gráficas de las variables que esten en memoria sin tener que saber el código para hacerlo.

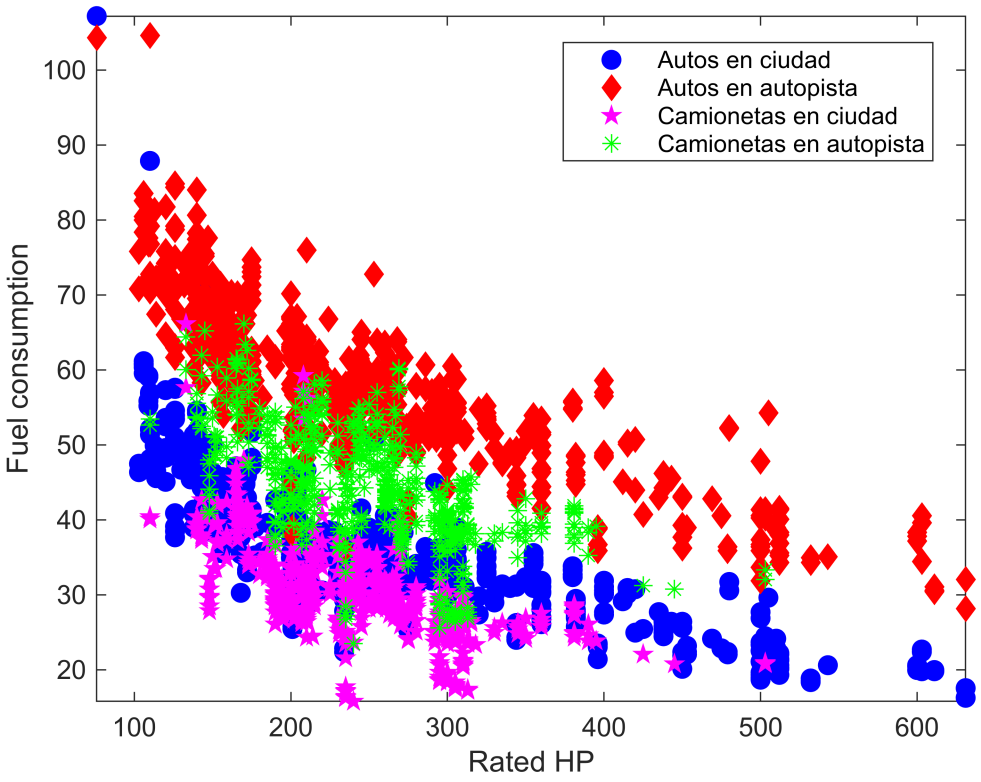
```
% Create scatter of selected data
s3 = scatter(data,"RatedHP","Km_gal","DisplayName","Km_gal");

% Add xlabel, ylabel, title, and legend
xlabel("RatedHP")
ylabel("Km_gal")
title("Km_gal vs. RatedHP")
legend

axis tight
grid
```



Visualización de los datos después de agruparlos

Podemos visualizar los datos agrupados de acuerdo a una categoría en una gráfica de dispersión usando gscatter. Note que hay "bandas" de información; podemos separar estas tendencias en varios conjuntos de datos para analizarlos por separado.

```
figure
z = gscatter(data.RatedHP, data.Km_gal,...
    {data.Car_Truck data.City_Highway},...
    "brmg","odp*",7,"on","filled");
legend(["Autos en ciudad" "Autos en autopista"...
    "Camionetas en ciudad" "Camionetas en autopista"])
xlim([min(data.RatedHP) max(data.RatedHP)])
ylim([min(data.Km_gal) max(data.Km_gal)])
xlabel("Rated HP")
ylabel("Fuel consumption")
```



Ajustar una ecuación personalizada

La dispersión de los datos en la gráfica inmediatamente anterior nos permite concluir que necesitaremos varios modelos por separado para ajustar cada segmento. De lo contrario nuestro modelo tendrá un ajuste muy probre. Vamos a automatizar la selección de los datos usando indexación lógica.

También, vamos a usar controles interactivos para facilitar la selección de los grupos.

```
Vehicle_type = "car";
```

```
City_highway = "highway";
Int_conf = 95;

grouping = data.City_Highway == City_highway & data.Car_Truck == Vehicle_type;
```

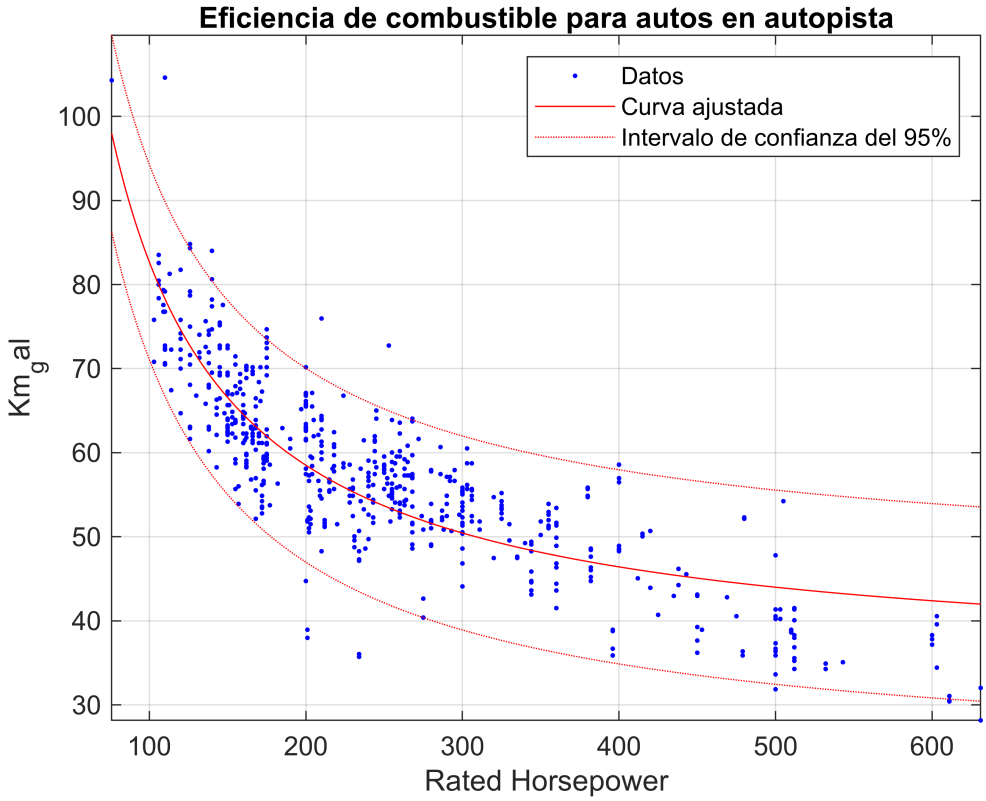
Aunque podríamos ajustar cualquier modelo matemático a nuestros datos, en este caso ya sabemos que las variables estan relacionadas de esta forma:

$$\mathrm{MPG} = b_1 + \frac{b_2}{\mathrm{RatedHP}}$$

Podemos usar esta ecuación para ajustar una curva personalizada usando la app Curve Fitter.

Usando la indexación lógica filtramos las categorias que deseamos ajustar y pasamos solo esta información a nuestra función generada por la app Curve Fitter.

```
[mdl, gof] = modelMPG(data,Vehicle_type,City_highway,Int_conf)
```




```
mdl =
     Linear model:
     mdl(x) = a + b*1/x
     Coefficients (with 95% confidence bounds):
       a =       34.33  (33.12, 35.55)
       b =        4836  (4597, 5074)
gof = struct with fields:
           sse: 2.3037e+04
       rsquare: 0.7033
           dfe: 669
    adjrsquare: 0.7028
          rmse: 5.8682
```