CO2 Emissions

Héctor Manuel Cárdenas Yáñez | A01634615

Siddhartha López Valenzuela | A00227694

Álvaro Morán Errejón | A01638034

Isaí Ambrocio | A01625101

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
from scipy.stats import norm, uniform, skewnorm

sns.set_theme()
```

```
df = pd.read_csv("/content/CO2 Emissions_Canada.csv")
```

```
df.head()
```

	Make	Model	Vehicle Class	Engine Size(L)	Cylinders	Transmission	Fuel Type	Fuel Consumption City (L
0	ACURA	ILX	COMPACT	2.0	4	AS5	Z	
1	ACURA	ILX	COMPACT	2.4	4	M6	Z	
2	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	
3	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	
4	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	

```
df.isna().sum()
```

```
Make                                0
Model                               0
Vehicle Class                       0
Engine Size(L)                      0
Cylinders                           0
Transmission                        0
Fuel Type                           0
Fuel Consumption City (L/100 km)    0
Fuel Consumption Hwy (L/100 km)     0
Fuel Consumption Comb (L/100 km)    0
Fuel Consumption Comb (mpg)         0
CO2 Emissions(g/km)                 0
dtype: int64
```

```
df.describe()
```

	Engine Size(L)	Cylinders	Fuel Consumption City (L/100 km)	Fuel Consumption Hwy (L/100 km)	Fuel Consumptic
count	7385.000000	7385.000000	7385.000000	7385.000000	
mean	3.160068	5.615030	12.556534	9.041706	
std	1.354170	1.828307	3.500274	2.224456	
min	0.900000	3.000000	4.200000	4.000000	
25%	2.000000	4.000000	10.100000	7.500000	
50%	3.000000	6.000000	12.100000	8.700000	
75%	3.700000	6.000000	14.600000	10.200000	
max	8.400000	16.000000	30.600000	20.600000	

Best Variable

> After testing with all possible variables, we noticed that the best was `Fuel Consumption City (L/100 km)` acording to R^2.

```
x_FCCity = df["Fuel Consumption City (L/100 km)"]
x_FCCity_const = sm.add_constant(x_FCCity)

y = df["CO2 Emissions(g/km)"]

model_FCCity = sm.OLS(y, x_FCCity_const)
result_FCCity = model_FCCity.fit()

print(result_FCCity.summary())
```

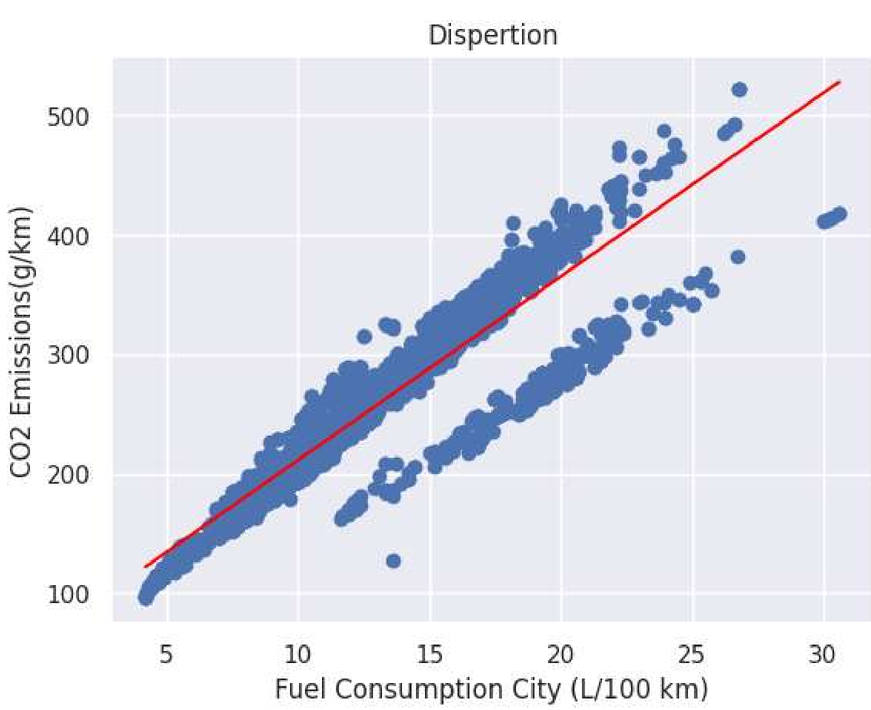
```
                            OLS Regression Results
==============================================================================
Dep. Variable:     CO2 Emissions(g/km)   R-squared:                       0.846
Model:                             OLS   Adj. R-squared:                  0.846
Method:                  Least Squares   F-statistic:                 4.045e+04
Date:                 Fri, 06 Oct 2023   Prob (F-statistic):               0.00
Time:                         23:02:47   Log-Likelihood:                -33630.
No. Observations:                 7385   AIC:                         6.726e+04
Df Residuals:                     7383   BIC:                         6.728e+04
Df Model:                            1
Covariance Type:             nonrobust
====================================================================================================
                                       coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------------
const                               57.5599      0.996     57.772      0.000      55.607      59.513
Fuel Consumption City (L/100 km)    15.3725      0.076    201.122      0.000      15.223      15.522
==============================================================================
Omnibus:                     3089.403   Durbin-Watson:                   1.913
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16424.392
Skew:                          -1.963   Prob(JB):                         0.00
Kurtosis:                       9.161   Cond. No.                         48.8
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
result_FCCity.params
```

```
const                               57.559903
Fuel Consumption City (L/100 km)    15.372459
dtype: float64
```

Scatter plot and trend line

```
B_0_FCCity = result_FCCity.params[0]
B_1_FCCity = result_FCCity.params[1]

# Crea una línea con los valores de B_0 y B_1
x_range_FCCity = np.linspace(min(x_FCCity), max(x_FCCity))
y_pred_FCCity = B_0_FCCity + B_1_FCCity * x_range_FCCity

plt.scatter(x_FCCity, y)
plt.title("Dispertion")
plt.xlabel("Fuel Consumption City (L/100 km)")
plt.ylabel("CO2 Emissions(g/km)")
plt.plot(x_range_FCCity, y_pred_FCCity, color="red") # Agrega la línea al gráfico
plt.show()
```

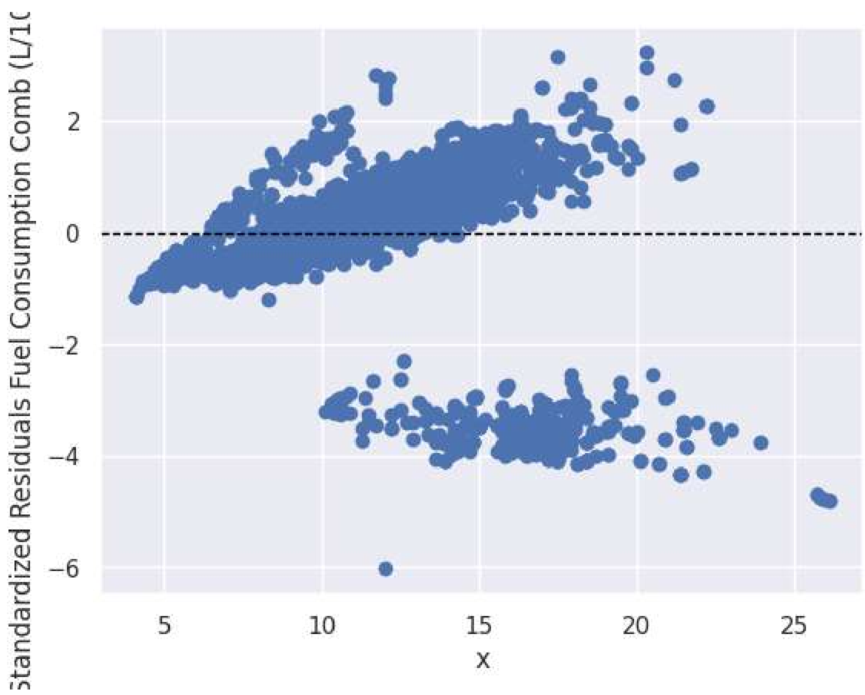


Studentized residuals

```
influence_FCCity = result_FCCity.get_influence()
standardized_residuals_FCCity = influence_FCCity.resid_studentized_internal

print(standardized_residuals_FCCity)
```

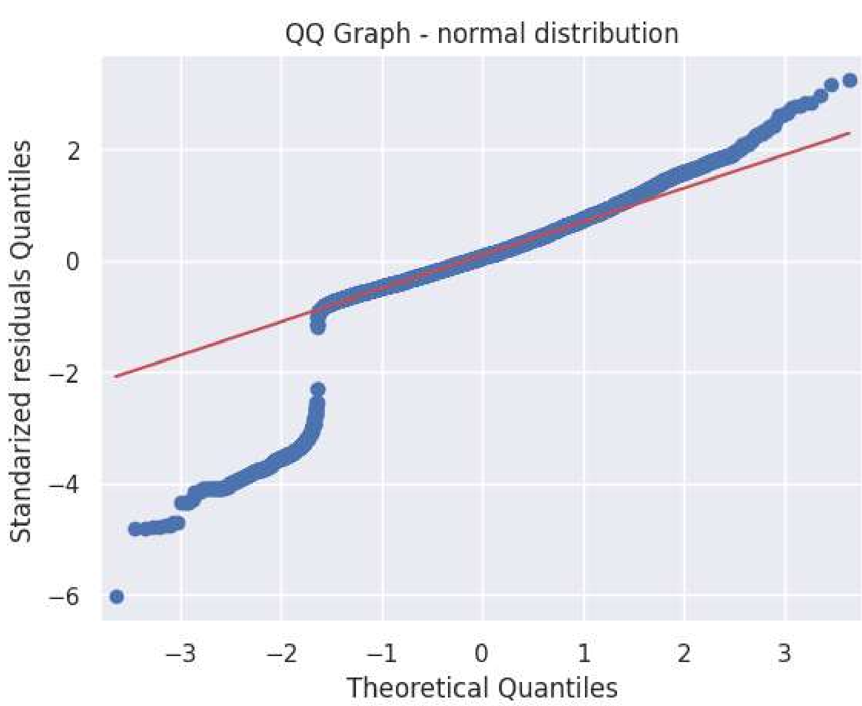
```
[-0.5980394  -0.37982853 -0.60022109 ...  0.11233374  0.09868503
  0.12598264]
```

```
plt.scatter(df["Fuel Consumption Comb (L/100 km)"],
            standardized_residuals_FCCity)
plt.xlabel("x")
plt.ylabel("Standardized Residuals Fuel Consumption Comb (L/100 km)")
plt.axhline(y=0, color = "black", linestyle="--", linewidth=1)
plt.show()
```



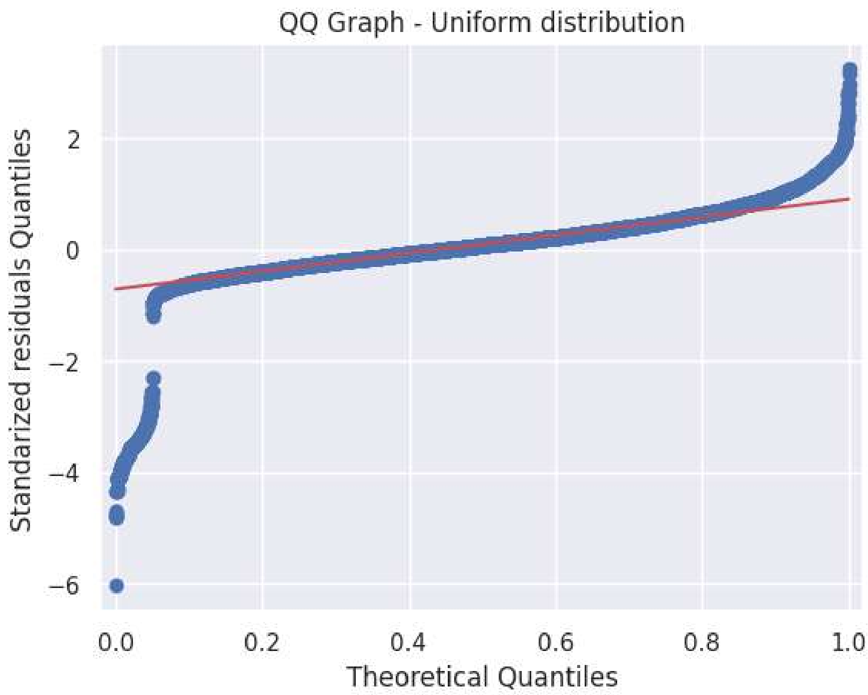
QQ graphs

```
fig = sm.qqplot(standardized_residuals_FCCity, dist = norm, line = "q")


plt.title ("QQ Graph - normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



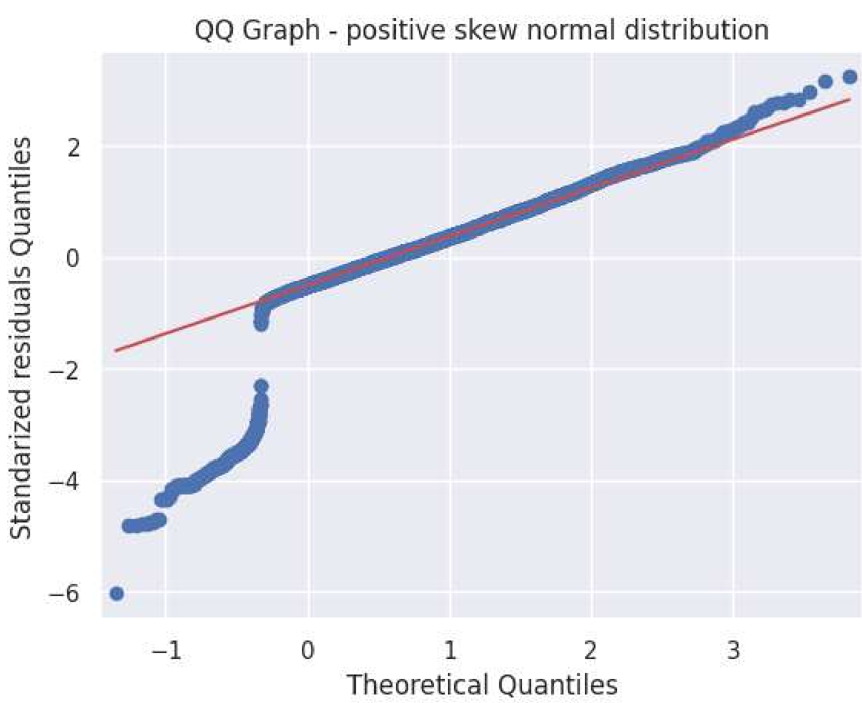
> Uniform distribution

```
fig = sm.qqplot(standardized_residuals_FCCity, dist = uniform, line = "q")

plt.title ("QQ Graph - Uniform distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



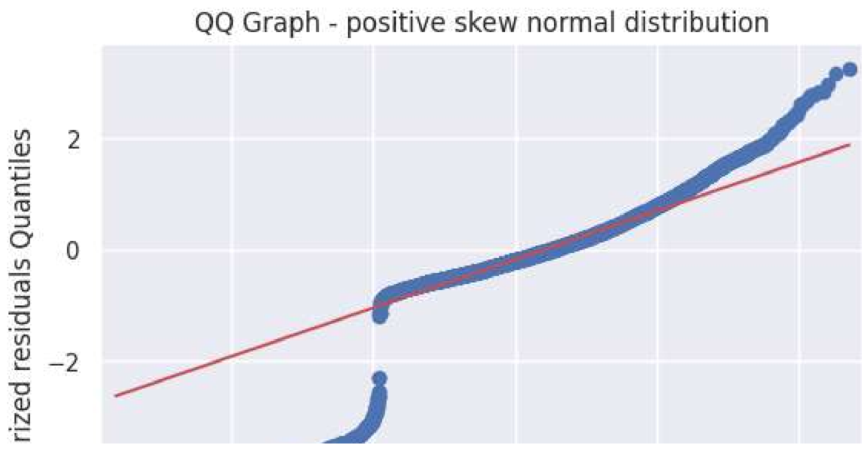
> Positive skew normal distribution

```
fig = sm.qqplot(standardized_residuals_FCCity, dist = skewnorm(2), line = "q")

plt.title ("QQ Graph - positive skew normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



> Negative skew normal distribution

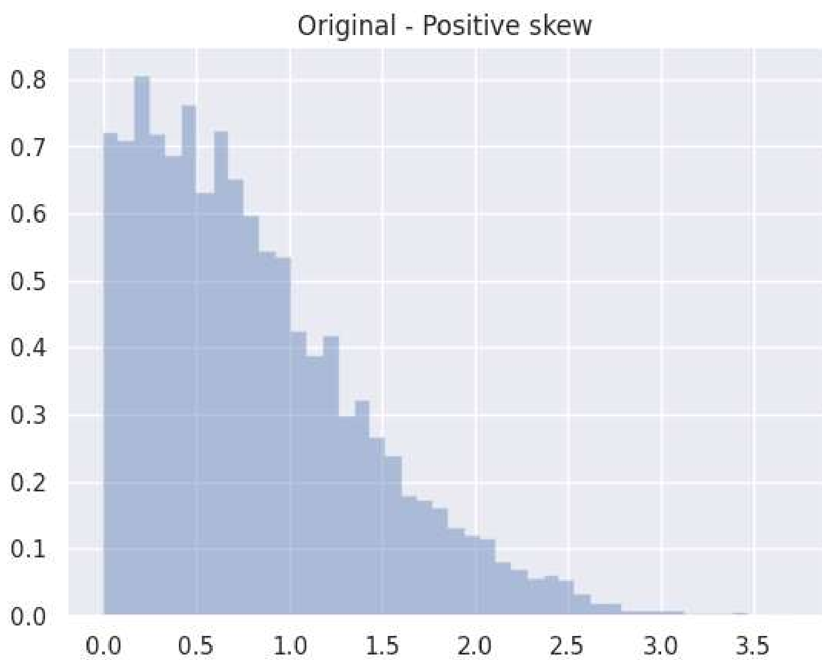
```
fig = sm.qqplot(standardized_residuals_FCCity, dist = skewnorm(-2), line = "q")

plt.title ("QQ Graph - positive skew normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



Transformación de la variable "y"(`CO2 Emissions(g/km)"`)

> Original - Positive skew

```
y_skew = skewnorm.rvs(y)
plt.hist(y_skew, density=True, bins="auto",
         histtype="stepfilled", alpha=0.4)

plt.title("Original - Positive skew ")
plt.show()
```



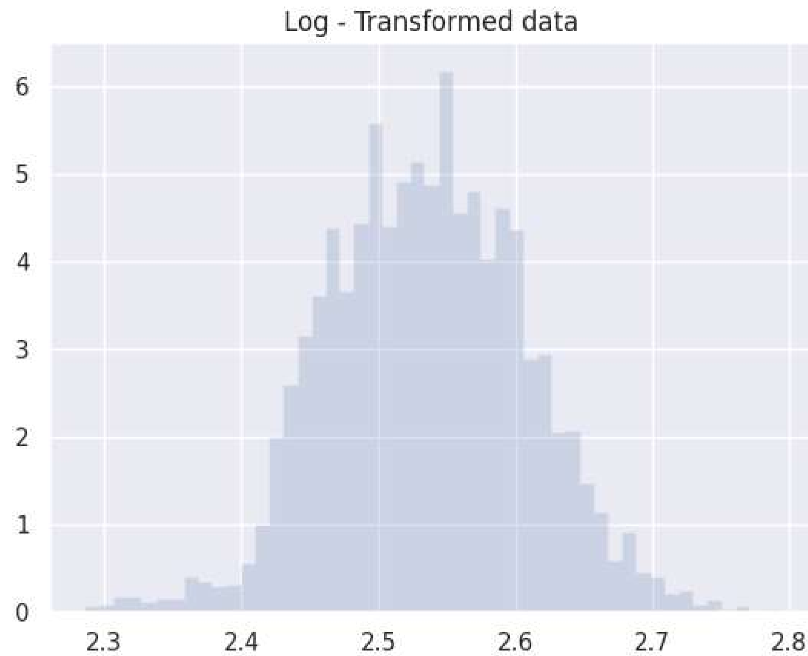
> Y squared - Transformed data

```
y_root = np.sqrt(y + abs(min(y)))
plt.hist(y_root, density=True, bins="auto", histtype="stepfilled", alpha = 0.4)
plt.title("Y squared - Transformed data")
plt.show()
```

Y squared - Transformed data

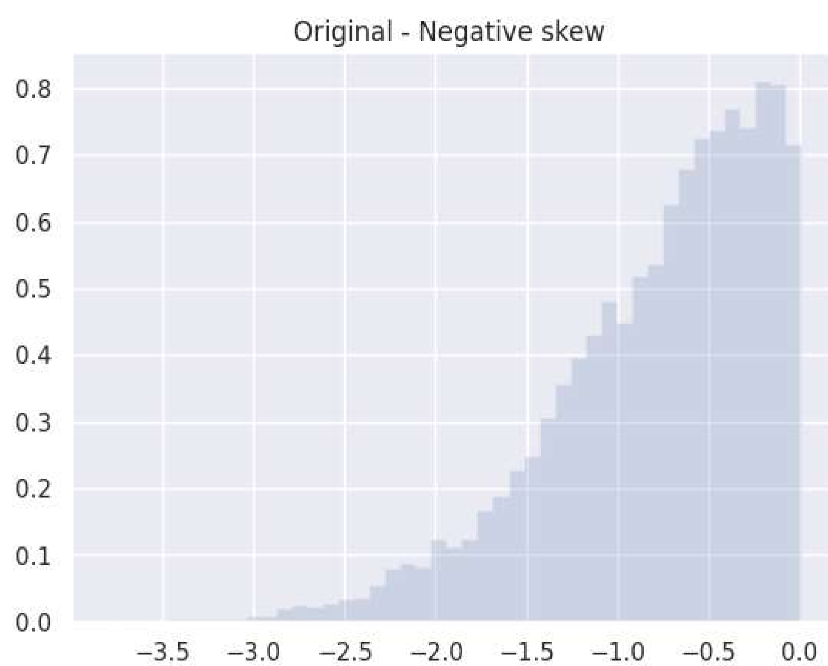
Log - Transformed data

```
y_log = np.log10(1 + y + abs (min(y)))
plt.hist(y_log, density=True, bins="auto", histtype="stepfilled", alpha=0.4)
plt.title("Log - Transformed data")
plt.show ()
```



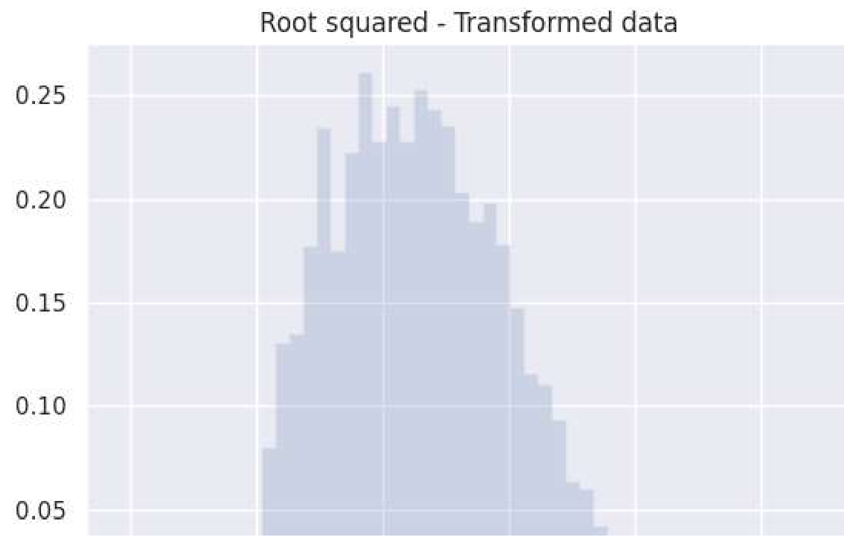
Original - Negative skew

```
y_neg_skew = skewnorm.rvs(-y)

plt.hist(y_neg_skew, density=True, bins="auto",
         histtype="stepfilled", alpha=0.4)

plt.title("Original - Negative skew ")
plt.show()
```



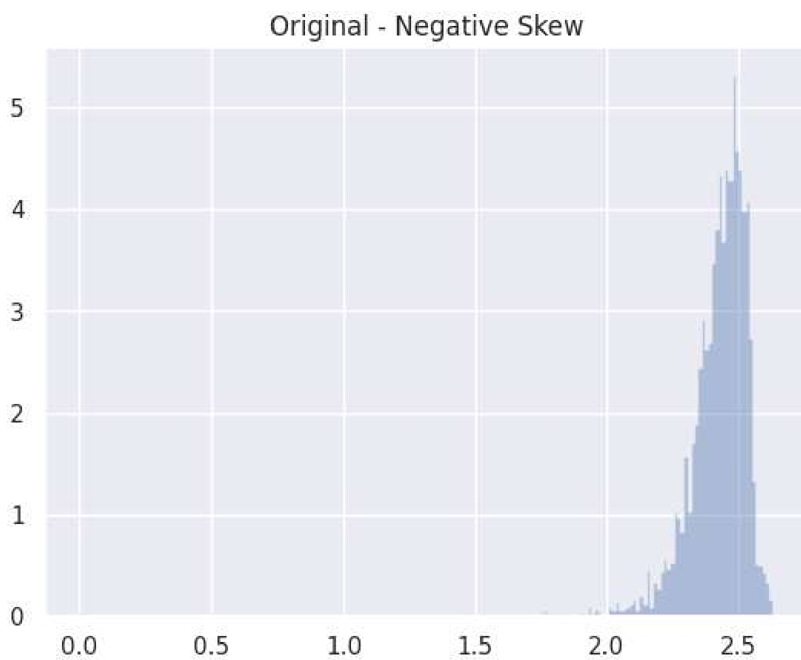
Root squared - Transformed data

```
y_positive = y + abs (min(y))
r_root = np.sqrt(max(y_positive) - y_positive)
plt.hist(y_root, density=True, bins="auto", histtype="stepfilled", alpha=0.4)
plt.title ("Root squared - Transformed data")
plt.show()
```



> Log - Transformed data

```
y_log = np.log10(1 + max(y_positive) - y_positive)
plt.hist(y_log, density=True, bins="auto", histtype="stepfilled", alpha=0.4)
plt.title("Log - Transformed data")
plt.show()
```



> From the graphs we can see that the best transformation for the variable `y` was root, because it is the one that is closest to a standard normal deviation.

Multiple Linear Regresion

```
x_complete = df[["Engine Size(L)", "Cylinders",
                 "Fuel Consumption City (L/100 km)",
                 "Fuel Consumption Hwy (L/100 km)",
                 "Fuel Consumption Comb (L/100 km)",
                 "Fuel Consumption Comb (mpg)"]]

x_complete_const = sm.add_constant(x_complete)
model_complete = sm.OLS(y, x_complete_const)
result_complete = model_complete.fit()
print(result_complete.summary())
```

```
                                  OLS Regression Results
==============================================================================
Dep. Variable:     CO2 Emissions(g/km)   R-squared:                       0.904
Model:                             OLS   Adj. R-squared:                  0.904
Method:                  Least Squares   F-statistic:                 1.157e+04
Date:                 Sat, 07 Oct 2023   Prob (F-statistic):               0.00
Time:                         02:58:41   Log-Likelihood:                -31880.
No. Observations:                 7385   AIC:                         6.377e+04
Df Residuals:                     7378   BIC:                         6.382e+04
Df Model:                            6
Covariance Type:             nonrobust
====================================================================================================
                                       coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------------
const                              227.8928      4.200     54.255      0.000     219.659     236.127
```

```
Engine Size(L)                           4.9936      0.456     10.962      0.000       4.101       5.887
Cylinders                                7.5385      0.319     23.657      0.000       6.914       8.163
Fuel Consumption City (L/100 km)        -0.0238      2.738     -0.009      0.993      -5.391       5.344
Fuel Consumption Hwy (L/100 km)          4.4906      2.260      1.987      0.047       0.061       8.920
Fuel Consumption Comb (L/100 km)         1.6730      4.969      0.337      0.736      -8.069      11.415
Fuel Consumption Comb (mpg)             -3.4235      0.079    -43.545      0.000      -3.578      -3.269
==============================================================================
Omnibus:                     1193.702   Durbin-Watson:                   1.618
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7810.498
Skew:                          -0.609   Prob(JB):                         0.00
Kurtosis:                       7.889   Cond. No.                         987.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

> In this model we use the y with no transportation. How ever, in the next one we use `y_root` and we noticed that our model has a small improved. It went from a `R-squared: 0.904` to a `R-squared: 0.915` as you can see below.

```
x_complete_transformed = df[["Engine Size(L)", "Cylinders",
                 "Fuel Consumption City (L/100 km)",
                 "Fuel Consumption Hwy (L/100 km)",
                 "Fuel Consumption Comb (L/100 km)",
                 "Fuel Consumption Comb (mpg)"]]

x_complete_transformed_const = sm.add_constant(x_complete)

model_trandformed_complete = sm.OLS(y_root, x_complete_transformed_const)
result_transformed_complete = model_trandformed_complete.fit()

print(result_transformed_complete.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     CO2 Emissions(g/km)   R-squared:                       0.915
Model:                             OLS   Adj. R-squared:                  0.915
Method:                  Least Squares   F-statistic:                 1.328e+04
Date:                 Sat, 07 Oct 2023   Prob (F-statistic):               0.00
Time:                         03:06:32   Log-Likelihood:                -4635.1
No. Observations:                 7385   AIC:                             9284.
Df Residuals:                     7378   BIC:                             9333.
Df Model:                            6
Covariance Type:             nonrobust
====================================================================================================
                                       coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------------
const                               19.4368      0.105    185.164      0.000      19.231      19.643
Engine Size(L)                       0.1331      0.011     11.687      0.000       0.111       0.155
Cylinders                            0.1856      0.008     23.303      0.000       0.170       0.201
Fuel Consumption City (L/100 km)    -0.0060      0.068     -0.087      0.931      -0.140       0.128
Fuel Consumption Hwy (L/100 km)      0.1182      0.056      2.093      0.036       0.007       0.229
Fuel Consumption Comb (L/100 km)    -0.0062      0.124     -0.050      0.960      -0.250       0.237
Fuel Consumption Comb (mpg)         -0.1191      0.002    -60.623      0.000      -0.123      -0.115
==============================================================================
Omnibus:                     1399.064   Durbin-Watson:                   1.617
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7822.151
Skew:                          -0.794   Prob(JB):                         0.00
Kurtosis:                       7.786   Cond. No.                         987.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

```
x_complete_backward = df[["Engine Size(L)", "Cylinders",
                          "Fuel Consumption Hwy (L/100 km)",
                          "Fuel Consumption Comb (mpg)"]]



x_complete_backward_const = sm.add_constant(x_complete_backward)


model_backward = sm.OLS(y_root, x_complete_backward_const)
result_backward = model_backward.fit()

print(result_backward.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:     CO2 Emissions(g/km)   R-squared:                       0.915
Model:                             OLS   Adj. R-squared:                  0.915
Method:                  Least Squares   F-statistic:                 1.991e+04
Date:                 Sat, 07 Oct 2023   Prob (F-statistic):               0.00
Time:                         03:06:23   Log-Likelihood:                -4636.1
No. Observations:                 7385   AIC:                             9282.
Df Residuals:                     7380   BIC:                             9317.
Df Model:                            4
Covariance Type:             nonrobust
```

```
==========================================================================================
                                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                         19.3676      0.092    209.826      0.000      19.187      19.549
Engine Size(L)                 0.1313      0.011     11.606      0.000       0.109       0.154
Cylinders                      0.1830      0.008     23.624      0.000       0.168       0.198
Fuel Consumption Hwy (L/100 km)  0.1078    0.005     19.768      0.000       0.097       0.119
Fuel Consumption Comb (mpg)   -0.1177      0.002    -70.478      0.000      -0.121      -0.114
==============================================================================
Omnibus:                     1452.486   Durbin-Watson:                   1.623
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             8021.114
Skew:                          -0.832   Prob(JB):                         0.00
Kurtosis:                       7.827   Cond. No.                         530.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

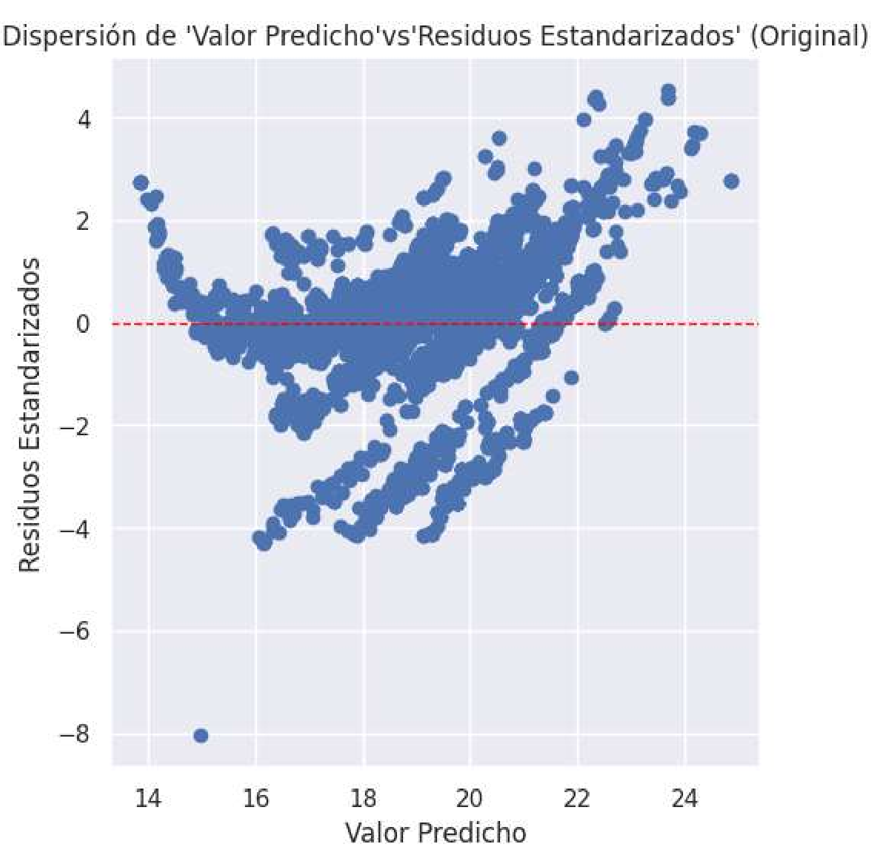
> As we can see, after removing the highest p values `Fuel Consumption City (L/100 km)` & `Fuel Consumption Comb (L/100 km)` the R^2 remains the same with less variables. That means that those variables are not significant for the model. So, the model is simpler and the model retains its `0.915` score *(Principle of parsimony)*. So, we accept the hypothesis.

Studentized Residuals

```
influence_backward = result_backward.get_influence()
standardized_residuals_backward = influence_backward.resid_studentized_internal

print(standardized_residuals_backward)
```

```
[-0.25076434 -0.06239508 -0.09337724 ...  0.48004602  0.58626641
  0.67521799]
```

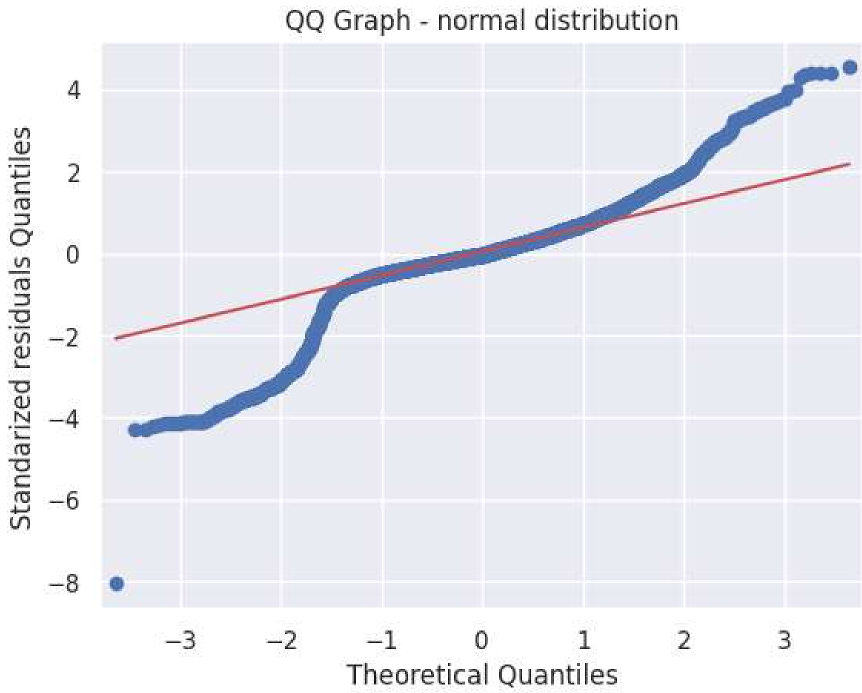
```
residuals = standardized_residuals_backward

# Gráfica de dispersión de "Valor predicho" vs "Residuos estandarizados" para
# los residuos originales.

plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.scatter(y_root, residuals)
plt.axhline(y=0, color='red', linestyle='--', linewidth=1)
plt.title("Dispersión de 'Valor Predicho'vs'Residuos Estandarizados' (Original)")
plt.xlabel("Valor Predicho")
plt.ylabel("Residuos Estandarizados")
plt.show()
```



> Normal distribution

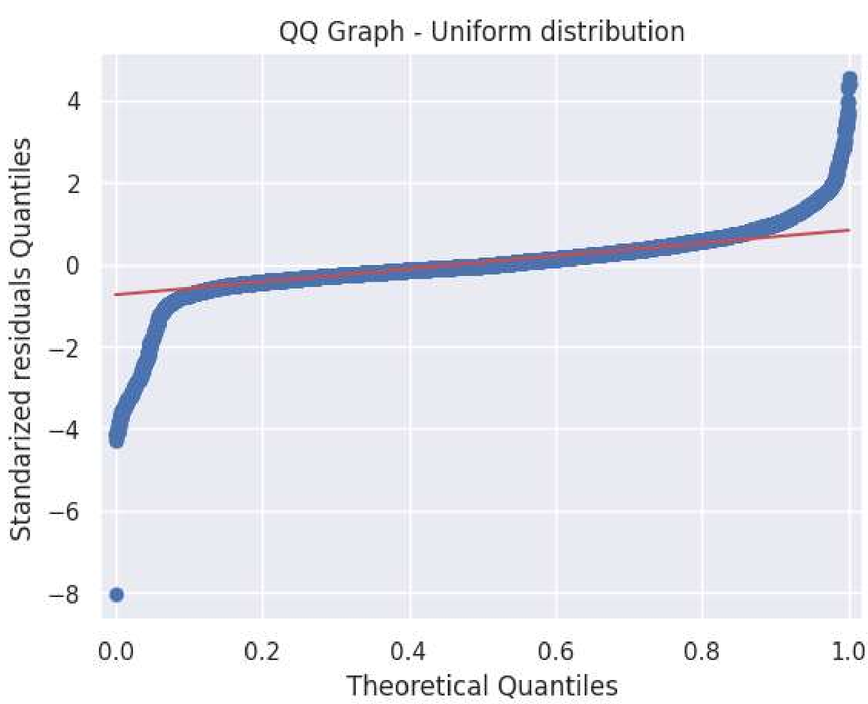
```
fig = sm.qqplot(standardized_residuals_backward, dist = norm, line = "q")
```

```
plt.title ("QQ Graph - normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



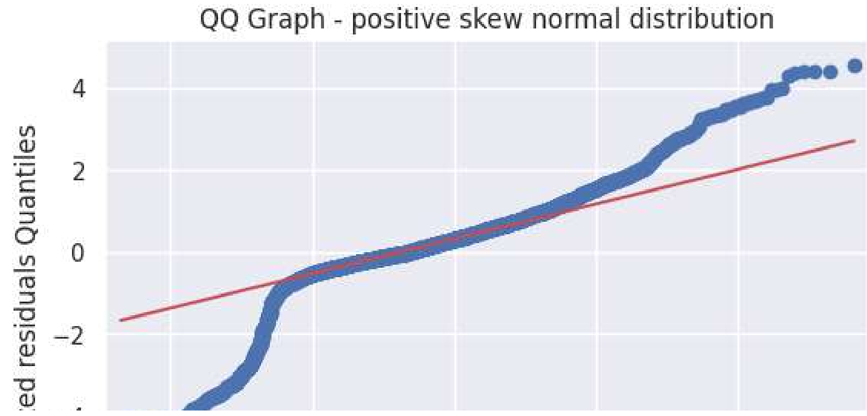
> Uniform distribution

```
fig = sm.qqplot(standardized_residuals_backward, dist = uniform, line = "q")

plt.title ("QQ Graph - Uniform distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



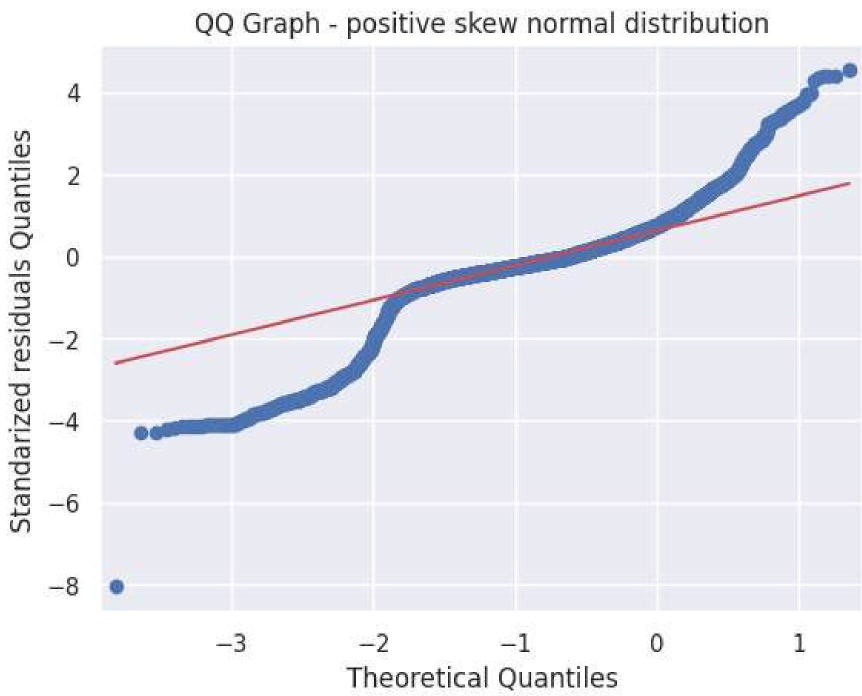
> Positive skew normal distribution

```
fig = sm.qqplot(standardized_residuals_backward, dist = skewnorm(2), line = "q")

plt.title ("QQ Graph - positive skew normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



Standarized residuals Quantiles

```
fig = sm.qqplot(standardized_residuals_backward, dist = skewnorm(-2), line = "q")

plt.title ("QQ Graph - positive skew normal distribution")
plt.ylabel("Standarized residuals Quantiles")
plt.show()
```



> As we can see, the distribution that align better with the line, is the `Uniform Distribution`. This indicates that the data have a uniform distribution. That is, all the values in the data set have approximately the same probability of occurring.