Does Your Car Consume a Lot of Gasoline?, Discovering With Regression Models

Pablo Eduado Duarte Tzuc
Data engineering student
(Universidad politécmica de Yucatán)
Km.4.5 carretera Mérida-Tetiz
Tablaje catastral 4448. Cp 97357
Ucú, Yucatán, México
Email: st1809063@upy.edu.mx

Abstract— **The following document works with the auto-mpg dataset that can be found in the UCI Machine Learning Repository or in the Kaggle community [3]. The objective in this dataset is to predict the variable "mpg" which refers to the consumption of gasoline per gallon. This work includes the problem understanding, data understanding, processing data and modeling with its results.**
In the problem understanding a more general view of the dataset is given, in the data understanding an light exploratory data analysis (EDA) is made to describe the data, explore it, verify the quality, and select the final dataset. Regarding the processing, the missing values were filled with the average, a logarithmic transformation was applied to the data set, on the other hand, some different approaches were tried to make it better, but in the end, they were not all good, so I use a simpler approach and finally the data were divided into the training and test set. Almost to finish, the k-fold was used as a strategy evaluation and linear regression models with and without regularization were used.

I. Introduction and Data Exploration

This dataset is from July 7, 1993, belonging to the StatLib library which is maintained by Carnegie Mellon University. The dataset was used in 1983 by the American Statistical Association Exposition. The data set that is currently used is slightly modified according to Ross Quinlan (1993), since 8 instances were eliminated because they did not contribute to the target variable (mpg).

It is composed of 398 instances with 7 input variables:

- Cylinders: Number of cylinders of car, multi-values discrete
- Displacement: Distance traveled by the car in miles, continuous variable.
- Horsepower: Automobile horsepower HP, continuous variable.
- Weight: Car weight in tons, continuous variable.
- Acceleration: Acceleration of the cars in m/s^2.
- Model year: Model year, multi-valued discrete.
- Origin: Manufactured for USA - Japan - Euro, 1,2 and 3 respectively.
- Car name: Name of the model (unique for each instance).

The main task of this dataset is predict the Fuel Efficiency:

- Mpg: Fuel consumption in miles per gallon, continuos variable.

The dataset has 6 missing values in the column horsepower.

The data frame looks like the following imagen.

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino

Using the pandas shape to verify the number of instances and columns, I verified that the dataframe has 398 instances and 9 columns.

One of the first things when starting to explore the data was to use the pandas info method to see if there are null values or if there are variables that are not properly in the type of value that they should, for example in this dataset the variable "horsepower" is found as a python object and should be a variable of type numeric to know if it can be used in the model.

Once the horsepower column became numeric, I used the pandas info() method again to verify that this was the case, but when applying it again, it was noticed that there were 6 missing values in that column as mentioned in the dataset description.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64
 2   displacement  398 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        398 non-null    int64
 5   acceleration  398 non-null    float64
 6   model year    398 non-null    int64
 7   origin        398 non-null    int64
 8   car name      398 non-null    object
dtypes: float64(4), int64(4), object(1)
memory usage: 28.1+ KB
```

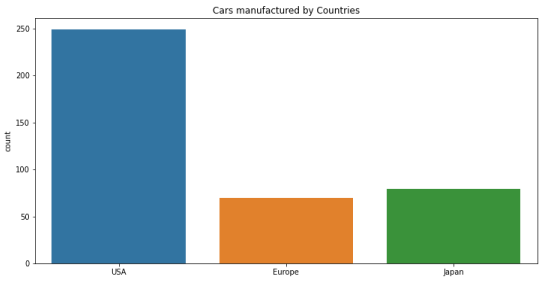
Use the pandas isnull() method plus sum to verify that it is missing values and it was, so I proceeded to find the mean of "horsepower" and have it replaced by the missing values so that there are no problems with data exploration.

The next thing I did was get simple statistics in the dataset, such as the mean, the standard deviation among others as shown below.

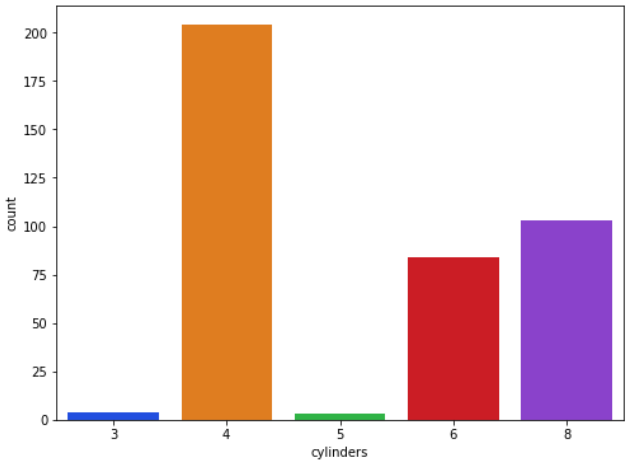
	count	mean	std	min	25%	50%	75%	max
mpg	398.0	23.514573	7.815984	9.0	17.500	23.0	29.000	46.6
cylinders	398.0	5.454774	1.701004	3.0	4.000	4.0	8.000	8.0
displacement	398.0	193.425879	104.269838	68.0	104.250	148.5	262.000	455.0
horsepower	398.0	104.469388	38.199187	46.0	76.000	95.0	125.000	230.0
weight	398.0	2970.424623	846.841774	1613.0	2223.750	2803.5	3608.000	5140.0
acceleration	398.0	15.568090	2.757689	8.0	13.825	15.5	17.175	24.8
model year	398.0	76.010050	3.697627	70.0	73.000	76.0	79.000	82.0
origin	398.0	1.572864	0.802055	1.0	1.000	1.0	2.000	3.0

As we can see in the previous image, there are three variables that have maximum and minimum by much difference, for example the minimum of the weight column is 1613 while the minimum cylinder is 3, so this can suggest a transformation in our set of data.

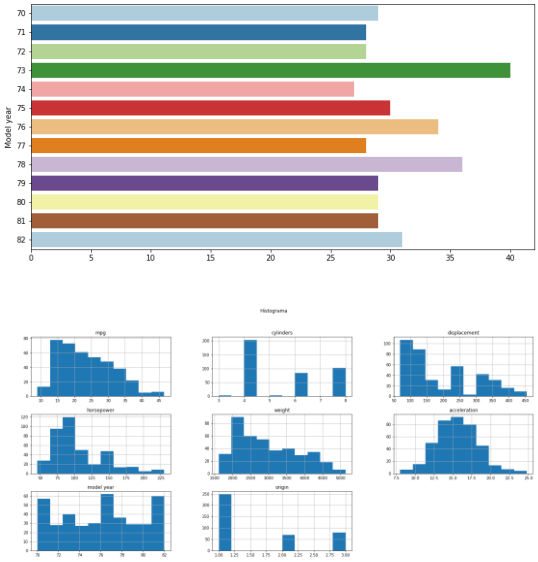
The next thing I did was to see the distribution of our target variable, in this image you can see that the mpg distribution is slightly skewed to the left, so one of my insight was to think about the possibility of applying a logarithm transformation to the dependent variable.



We can see that most of the cars in the dataset contain 4 cylinders and in second place are those with 8 cylinders, the greater the number of cylinders, the greater the gasoline consumption [5].

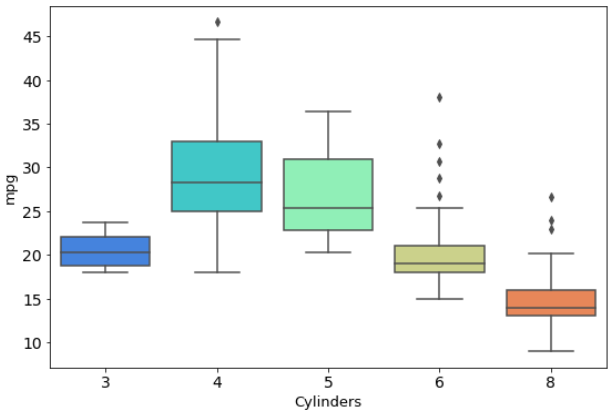


In this graph we can see that in the data set most of the cars are from the year 1973, well, they already have time, will the year of a car influence the consumption of gasoline? Maybe yes, maybe not, we can check that thanks to a correlation matrix, which will be shown later.

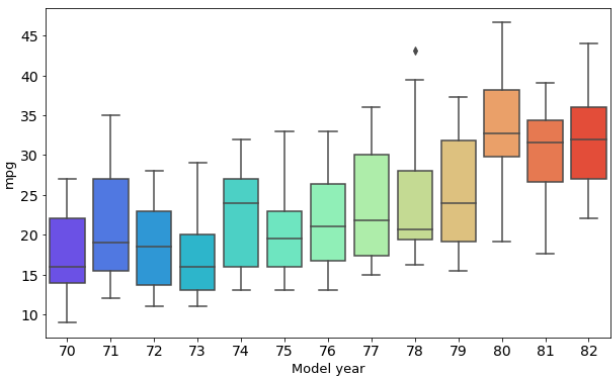


In the graph above we can see the distributions of our variables and it can be seen that the mpg, displacement, horsepower and weight distributions are slightly skewed to the left. This suggested applying a transformation to said data, creating new variables that can be used for my model and give it more added value.

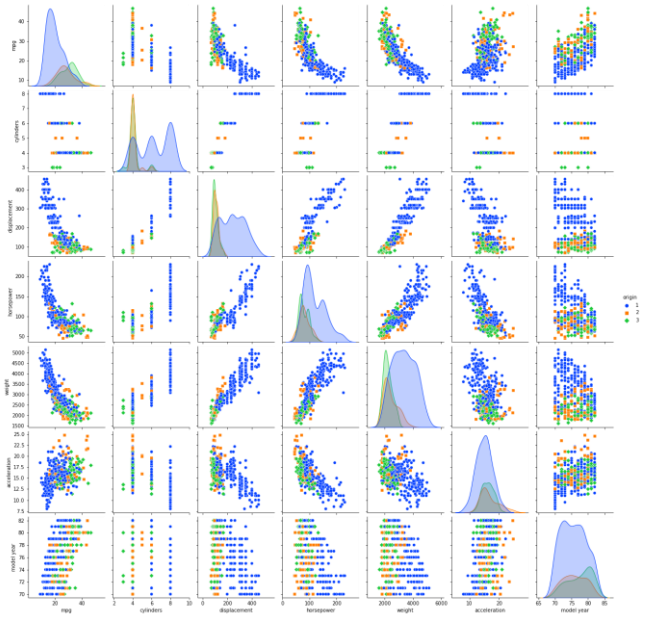
In the following graph of mpg with respect to cylinders use boxplots to see possible outliers, we can see that there are not a lot of outliers, so they can be left that way for now.



As in the previous image we use boxplots to see possible outliers, but this time the mpg variable with respect to the year of the car.

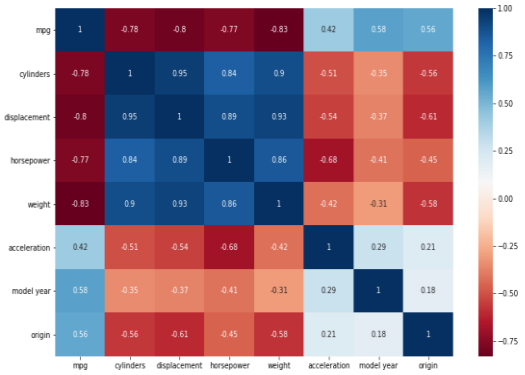


Thanks to this graph we can see the relationships that some variables have with others, as we can see most of the relationships between them tend to have a negative or positive correlation, but they are not widely dispersed among them. It can also be seen that the data pertaining to Japan (green color) and Europe (orange color) are slightly skewed to the left.



From my point of view with the previous graph I thought that a multilinear model would be one of the most effective to use in this data set.

To finish this part, use a correlation matrix like the one below.



The correlation matrix above can give us very interesting insights, which can leave us wondering or at least it is what happened to me. For example, we can see that the number of cylinders and gasoline consumption have a negative correlation, that is, that major number of cylinders, the gasoline consumption decrease, for me this was doubtful because I know that cars with a greater number of cylinders, they consume more gasoline, so I began to investigate and several pages said that the greater the number of cylinders, the greater the gasoline consumption (Fernando Garcia, 2017)

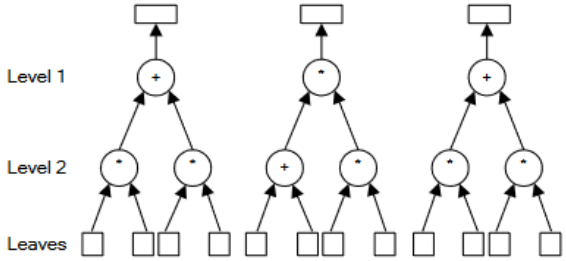
A positive correlation in which it made a lot of sense to me was that of weight and horsepower, because the more weight a car has, more horsepower it needs.

On the other hand, with the correlation matrix, we can try to remove variables that could be redundant, for example it can be seen that the variable "displacement" with the number of cylinders has a high positive correlation. We could consider a redundancy of 90% and remove those variables that have a greater correlation of 90%, this was one of the approaches used for the modeling part, we will talk about it later.

II. Previous Work

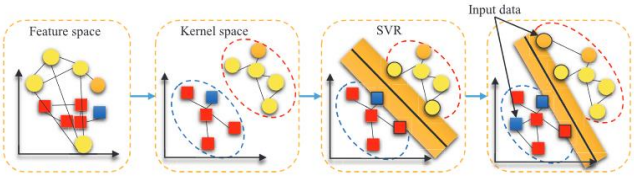
One of the ways in which the mpg value was tried to be predicted is with the help of three architectures from Fuzzy neural networks [4], the paper does not mention any robust preprocessing or special data methodology, Chang-Wook Han gives to understand that the data used is the 392 instances and ignoring the column "car name", the data used as input variables is numerical data only like the variable to predict (mpg).

In the following image you can see how the architecture of Fuzzy neural networks looks like.



In the focus of this paper, the holdout method was not used, 50% of the data was used for the training set as well as the other 50% of the data for the test set. The paper does not use the RMSE or R ^ 2 for the verification of the results, instead it uses population size, generation size, crossover rate, mutation rate and learning rate. Overall, the paper says that Tree architecture of fuzzy neural networks have an advantage of reducing the number of rules by selecting fuzzy neurons as nodes and relevant inputs as leave so optimally.

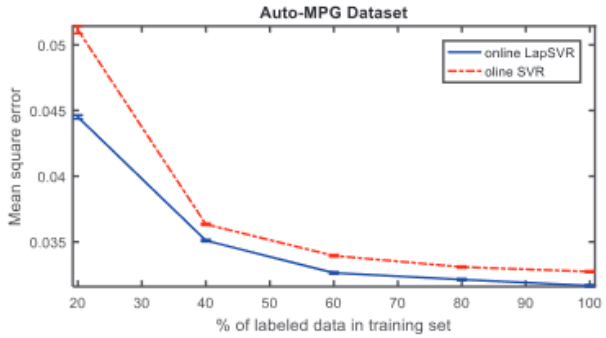
In the paper titled as "Online Laplacian-Regularized Support Vector Regression" he uses a new approach that uses support vector regression (SVR) in a multiple regularized frame, named as inline SVR regularized by Laplacia or LapSVR online avoids calculations repeating of batch methods, requiring intensive calculations, especially with massive size or dimension data.



In this paper 3 datasets were used in which the auto-mpg one is found, all datasets were scaled between 0 and 1 and All experiments use the $\varepsilon - insensitive$ loss function and the Gaussian RBF kernel K.

Of the auto-mpg data sets, 65 percent were used for the training set and 35 percent for the test set. As such, the paper

only mentions that the data is staked from 0 to 1, it does not say that it has used any other transformation or modification, it is also mentioned that 8 vectors were used, so it can be inferred that the ignored variable was that of " car name "because it is the only one of type string. The training set was labeled with a rate from (0.2, 0.4, 0.6, 0.8, 1.0), and run the train and test processes five times, and record the test results.



The results in this paper are shown as in the previous graph, where the online SVR and its approach is compared, that is, the online LapSVR.

The results confirm that the proposed algorithm offers better performance than the traditional online SVR algorithm and approaches the supervised learning method as the iteration time increases [1].

III. Data Preprocessing and Modelling

Regarding this part of the document, as previously mentioned the 6 missing values of the horsepower column were filled in with the mean, then try to apply different perspectives or approaches and see which of them gives me a better model, one of the first approaches What I tried was to want to create new columns that would generate more value to the model, I created two new columns the first was the ratio between weight and acceleration the second was the ratio between the variable 'displacement' and cylinders, however when dividing my data set at 80 percent for the training set and 20 percent for the test set, both my RMSE and R ^ 2 were not entirely satisfactory so I proceeded to discard this approach.

The second approach I used was to simply create the input variables as the data was found, that is, without any transformation, but if they are valid to make the regression models, the dataset was divided using the hold-out method [6] and the results obtained were not bad, but thanks to the data exploration previously done I had in mind that this could be improved by applying some transformation to the data, but I wanted to leave this approach at the end to continue treating others. In the following image we can see the RMSE and R ^ 2 for both the training set and the test set using the second approach.

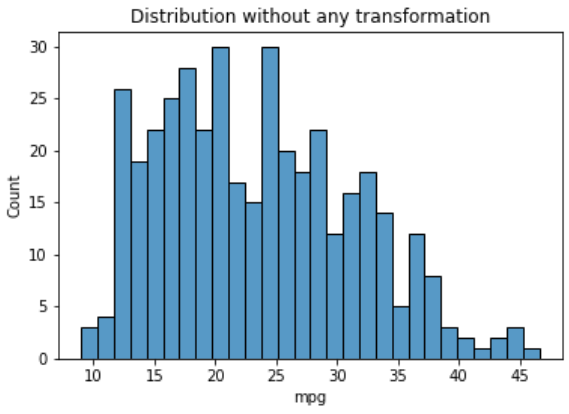
```
print(f'Test set RMSE = {rmse_test}')
print(f'Train set RMSE = {rmse_train}')

print(f'\nTest set R2 = {R2_test}')
print(f"Train set R2= {R2_train}")
```

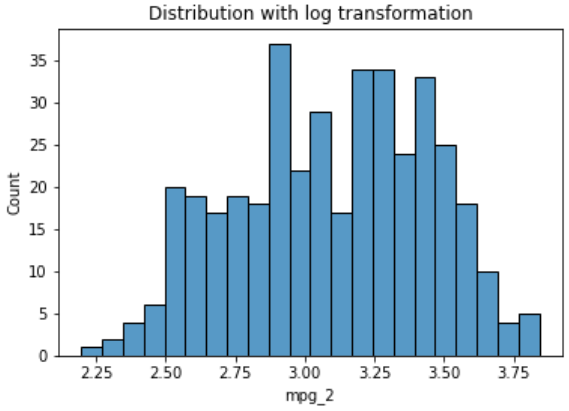
```
Test set RMSE = 3.474759231090624
Train set RMSE = 3.295291937017796

Test set R2 = 0.8051056783891937
Train set R2= 0.8208994581849991
```

As we can see, the R ^ 2 for the training and testing set is not very different, so our model is not overfitted, this may indicate that using a regression with regularization such as Lasso or Ridge does not improve the model at all, although this will be shown later.

The third approach that I used was to make a reduction of the variables that are redundant or that contribute less to the model, so I used SelectKBest with score_fun for regression, of the 7 input variables that we had I wanted to obtain the 5 best, but unfortunately when doing this the model with linear regression got worse, so I tried the same, but using the best 6 input variables, the result was that there was an improvement but it was not better than the second approach where the 7 input variables were used, so I discarded variable filtering.

Finally, I decided to make a transformation to the data, since I had observed the distributions of the variables and especially the target variable, I had in mind that this approach could be better, so first transform the target variable (mpg) using logarithm, that is to say apply a logarithmic transformation, as you can see below, we can first see the distribution without the transformation and then with the corresponding transformation.





Apparently, everything was going along the way, the problem that I had now was to decide whether to transform all the variable inputs, only some or even none, in my opinion the variables that should be transformed as well as the target were:

- Displacement
- Horsepower
- Weight

For its distributions seen before. As such, I did not know, so I began to investigate, but unfortunately, I could not find something that was concrete and concise, so I decided to also transform all the variable input since it caused me noise only to transform the target variable and some input variables.

When using this approach with multiple linear regression my RMSE and R ^ 2 were good and even better than the second approach, so in effect the model is better applying the logarithmic transformation to the data set.

Something important to mention is that I use 25% of the data for the test set, the remainder for the training set, because the dataset is not too large [2].

So now that I have the right data and when applied in LinearRegression () the RMSE and R ^ 2 were good, the next thing was to apply the multiple linear regression with regularization, so I first proceeded to use Lasso regression with cross validation (cv=5) and the results were as follows.

```
RMSE =  0.11708346672166832
R^2 =  0.8889916082516586
Alpha =  0.0001
```

Then the next regression with regularization that I used was Ridge regression with a cross validation of 5 as in Lasso regression, the results are as follows.

```
RMSE =  0.11708346672166832
R^2 =  0.8889916082516586
Alpha = 0.0001
```

As we can see the results are identical, so we can also infer that using a regression with regularization is not adequate.

Finally use ElasticNetcv () with the following parameters:

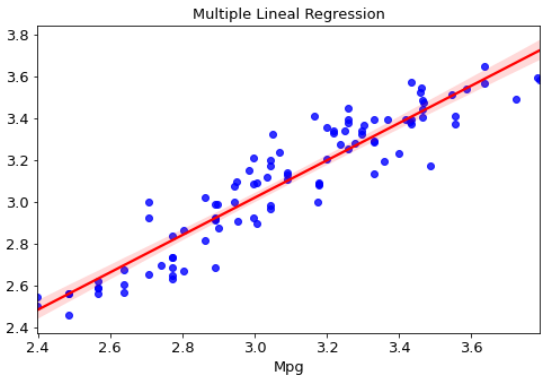
```
model_elasticnetcv = ElasticNetCV(
    alphas = [0.1, 0.01, 0.005, 0.0025, 0.001],
    l1_ratio = [.01, .05, .25, .5, .75, .95, .99],
    cv = 5
)
```

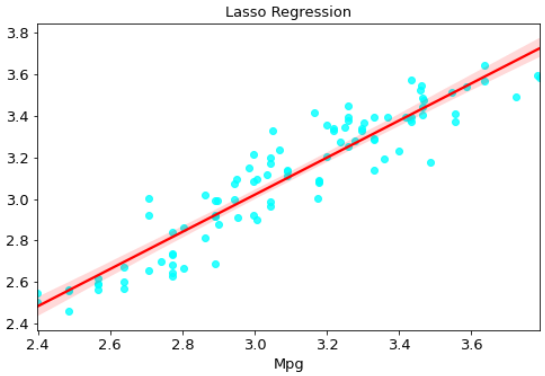
As we can see, a cross validation of 5 was also used and the results obtained were the following:

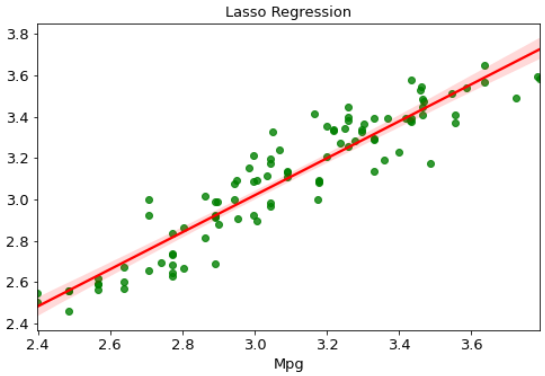
```
alpha = 0.001
l1 ratio = 0.99
RMSE = 0.12495738987845816
R2 = 0.8713950090783882
```

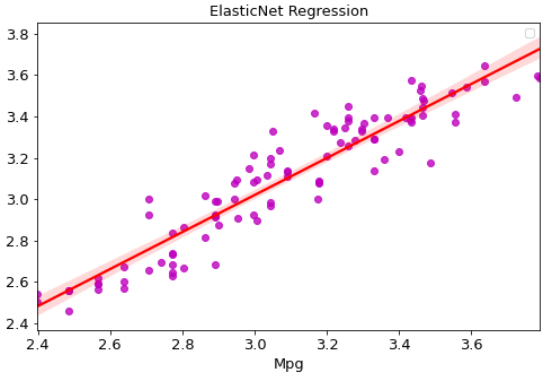
As we can see, alpha is close to zero and l1 is almost one, so we can conclude that making a linear regression model with regularization would not be a correct approach in the next part I will make a comparison to see this more clearly.

IV. RESULTS









Model	RMSE	R^2	Alpha	L1_ratio
Multiple lineal regression	0.1170	0.888	-	-

Lasso regression	0.1170	0.888	0.0001	
Ridge regression	0.1170	0.888	0.0001	
ElasticNet regression	0.1249	0.8713	0.001	0.99

As we can see the graphs are very similar or equal to Multiple linear regression, because the values of alpha in ridge and lasso regression are closely to cero and in the case of the regression with ElasticNet alpha is close to cero and L1_ratio is close to one, so it means that the regularizations aren't necessary, therefore the best approach is the multiple linear regression.

V. Conclusion and future work

As we could see in the previous work, all the previous steps before applying any machine learning model are important, since it is necessary to obtain a better perspective on the data with which we are working and to know what type of approach is the best or suitable that can be applied to our model. Also, as it was possible to see the use of graphs and visualizations, we can have a better understanding about possible approaches that can be given to the data, in this case thanks to the visualization of the histogram of the objective variable we can realize that the distribution was slightly skewed to the left so using a logarithmic transformation could help us to have a better model. In this dataset I tested 3 different approaches to obtain a better model, so it is to go through trial and error in the ideas that we come up with to obtain a better model from my perspective, of course doing trial and error once we have a larger overview with the steps before preprocessing data.

On the other hand, it is important to know and know the evaluation metrics and theory behind for machine learning models, since without them we can go with the pretense that we have a good model when it is not like that, currently I have not yet fully mastered this part so that in future works I would like to delve more deeply into these aspects, because as I said they are indispensable. Undoubtedly, the dataset with which I work is old, so it would be interesting to do the same type of work, that is, to predict gasoline consumption but with current cars and make a comparison, to see how certain aspects or characteristics of the vehicles have changed. cars. Since the technology and mechanics in a car from 30 or 40 years ago is very different from those of today or even one from 2010.

References

[1] L. Zhang and W. Liu, "Online Laplacian-Regularized Support Vector Regression," *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, Exeter, 2017, pp. 1-6, doi: 10.1109/CYBConf.2017.7985796.

[2] O. W. Samuel, «Researchgate, » 2016 March 18. [En línea]. Available: https://www.researchgate.net/post/Is-there-an-ideal-ratio-between-a-training-set-and-validation-set-Which-trade-off-would-you-suggest.

[3] U. M. L. Repository, «UCI Machine Learning Repository, » 7 July 1993. [En línea]. Available: https://archive.ics.uci.edu/ml/datasets/Auto+MPG.

[4] HAN, Chang-Wook. Auto MPG Prediction using Tree Architectures of Fuzzy Neural Networks. 2017.

[5] Ross, Marc. "Automobile fuel consumption and emissions: Effects of vehicle and driving characteristics." *Annual Review of Energy and the Environment* 19.1 (1994): 75-112.

[6] J. Brownlee, «Machine learning mastery, » 26 August 2020. [En línea]. Available: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/.