EDA and Prediction of International Coffee Prices with a Variety of Algorithms of Machine

Simon Atehortua. ID 501094927

Supervisor: Ceni Babaoglu, Ph.D.

October 24, 2022

Ryerson University

Table of Contents

Links **3**

GitHub Link 3

Data Set Link 3

Abstract **4**

Literature Review **6**

Overall Methodology **9**

Diagram of an overall methodology 9

Data Acquisition **10**

Columns Metadata 12

Multivariate Analysis **14**

Summary Statistics 14

Min, Max, and Outliers 16

Boxplot or candle sticks 17

Frequency Distribution 18

Correlation and Heatmap 20

Exploratory Data Analysis **23**

Features Selection and Backward Pairwise Relations **26**

Normalization **30**

References **33**

Links

GitHub Link

https://github.com/montwa/Ryerson

Data Set Link

https://federaciondecafeteros.org/app/uploads/2020/01/Precios-%C3%A1rea-y-producci%C3%B3n-de-caf%C3%A9-3.xlsx

Excel Sheet: 6. Precio OIC Mensual

Abstract

Coffee is one of the greatest drinks that humanity could have ever discovered. The prices of coffee are expressed in US cents per pound of green coffee and many factors affect the prices. The ICO (International Coffee Organization), will collect daily prices from the New York, Germany, and France futures exchanges and will set up a price based on these factors and a variety of conditions.

The idea of this project and after doing an exploratory analysis of the data is to predict the prices of the ICO market given the prices in Europe, more exactly in the countries of Germany and France, as well as in the United States of America of a different variety of Coffee that is grouped as follows: Colombian Mild Arabicas (Colombian Excelso UGQ screen size 14, Colombian Excelso European preparation screen size 15), Other Mild Arabicas (Costa Rica hard bean, Mexico Prime washed, Honduras high grown, Guatemala prime washed, El Salvador Strictly High grown, Guatemala hard bean, Honduras High grown European preparation) Brazilian (Brazil Santos ¾ screen size 14/16, Brazil Santos ⅔ screen size 17/18, Brasil Santos ¾ screen size 14/16) and Robustas (Vietnam grade 2, Indonesia EK grade 4, Uganda Standard, Côte d'Ivoire grade 2).

All data for this project comes from Federación Nacional De Cafeteros de Colombia (The Colombian National Coffee Growers), which is the only authorized entity by the government of

Colombia, that is responsible for exporting, buying, and setting the prices of coffee, the website (https://federaciondecafeteros.org/wp/coffee-statistics/?lang=en), section Coffee prices, area, and production have an excel file with the data required during this project. The data will be cleaned and checked for any inconsistencies or errors that can alter our results, after that, there will be an Exploratory Data Analysis to check the behavior of the time series data, and to possibly discover new information that could help find important and relevant information about our study. Through a variety of different Machine Learning regression algorithms like linear regression, SVM (Support Vector Machine), KNN regressor (K-Nearest Neighbors), Decision Tree regressor, and Neural Networks, I will intend to predict the future price of the coffee set by the International Coffee Organization.

A variety of tools will be used for this project: Jupyter Notebook, Python, R studio, Microsoft Excel, Github, and Google Documents.

During this project, a variety of questions have to be formulated in order to obtain the best answer to our results. An example of these questions, among many others, are: which is the most accurate algorithm to predict the ICO prices of coffee and why the selection of this algorithm? What will be the future price in the short term given that the model has been adequately trained and formulated? Why is this investigation relevant to the student and how it can help to develop his knowledge not only in exploratory data analysis but as well in machine learning algorithms?

Keywords: coffee, exploratory data analysis, regression analysis, prediction, machine learning algorithms

Literature Review

Coffee has been the most important agricultural product in Colombia, providing livelihood for around two million Colombians (Sanz Uribe et al., 2021, 35). Linear regression has many applications in life, one of them, and the one that I am going to be applying is predicting the price of coffee based on different inputs or variables, this has been already tried by a variety of people using different linear models, Neural Networks, and Support Vector Machines, like in a case study made in India, to forecast Indian green coffee, a study made in Brazil to predict as well the value of the grain using Neural Networks, (Deina et al., 2021, 2)

But why would we want to predict the coffee price? What would be the advantage of doing this? Well, just imagine if coffee growers had been prepared by 2020 when the price of coffee was the highest on the New York stock exchange at US$1.10 per pound instead of selling the coffee in the harvest of 2019 when prices were below US$ 1 per pound, (Diario el Mundo, 2021), this fluctuation in prices could have been a great opportunity to growers, and perhaps if is seen from the client or purchaser side, it could be argued that they would want to buy at the lowest prices, creating perhaps a balance in the economy as both ends won't be able to take advantage from each other.

One important factor to be analyzed would be the market size and the demographic impact on different societies around the world, with emerging markets like China, India, and Brazil that constitute 36% of the global GDP (Li et al., 2022, 1) is in these countries where tea consumption is superior to coffee, with the exception of Brazil, and is in these countries where customers would pay a premium price for the symbolic western experience at stores like Starbucks (Li et al., 2022, 2) just for a cup of coffee.

Just imagine for a moment, if the regression analysis to predict the prices is combined as well with an algorithm for coffee selection based on its quality using different techniques like k-Neighbours, Decision Tree, SVM Support Vector Machine, Logistic regression, or Neural Networks, and having these results or classification of superior quality as independent variables in the price regression, and not depending solely on other factors that are not quality, luckily there are new studies that are already doing the first step, like Suarez Pena, on his thesis, to classify based on quality, the results having a prediction of 83% in classification, and Mean Absolute Error (MAE) of 14,61% (Suarez Pena, 2019,100), this would be a big tool not only for coffee tasters but growers and purchasers as well as people who really enjoy drinking quality coffee.

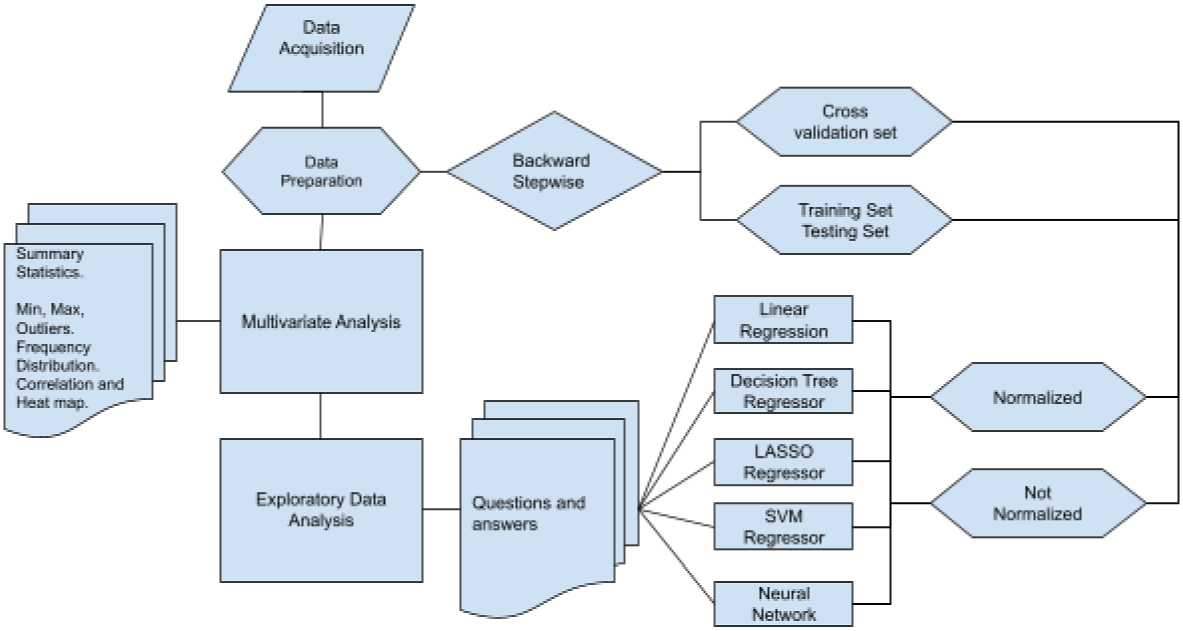
‘Professional investors favour two dominant schools of thought on investing which are Fundamental Analysis and Technical Analysis, and Machine learning techniques are one of the

Technical Analysis, (Siew et al., 2012,2), and some of these algorithms are the ones that I will be using for the project, using examples and getting help from some of the papers published in the scientific community (Sreehari et al., 2018,1) where the author uses Multiple Linear Regression to predict climate, another very interesting article where they use regression tree to predict the Indonesian stock price during the Covid-19 era (Hindrayani, et al.,2020,1), as well as a comparative analysis on linear regression and Support Vector regression (SVR)(Kavitha S, et al.,2016,1) where they analyze a time series data in order to have better prediction and accuracy on their data. Which is the most accurate algorithm to predict the future price of the OIC_prices? The need to choose a different algorithm varies according to the needs of each investigation (Gil Serna, 2012, 82), however the final decision, in this case, will be made once the results of the analysis have concluded.

The project is very important for the student as it opens new fields of thinking in the machine learning and Data Science world, and encourages him to do his own research about topics that the student has never seen or taken and is because of research that new things are discovered.

Overall Methodology

Diagram of an overall methodology



After acquiring the data and preparing it, will be used to extract a summary of Statistics and do an Exploratory Data Analysis in order to answer possible questions that might arise, then the data will be used with 2 algorithms Training Set and Testing Set, and cross-validation, then Algorithms like Linear Regression, Decision tree Regression, Lasso Regressor, SVM Regressor, and Neural Network will be used to answer the questions that were formulated in the abstract and to see how our regressors are behaving

When data needs to be normalized, it will be normalized with Standard Normalizer before applying the algorithms previously mentioned above.

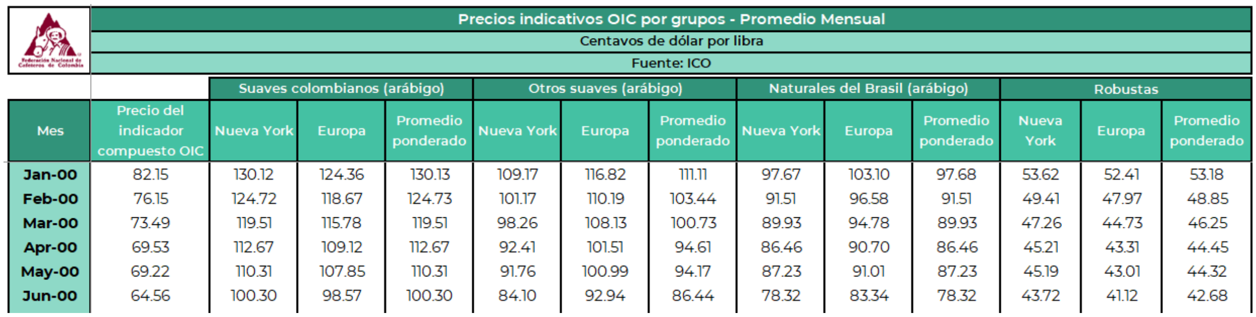
Data Acquisition

The source of the dataset, can be downloaded from the website of the Federación Nacional de Cafeteros de Colombia (National Federation of Coffee Growers of Colombia) here:

https://federaciondecafeteros.org/app/uploads/2020/01/Precios-%C3%A1rea-y-producci%C3%B3n-de-caf%C3%A9.xlsx

Even if it contains a variety of sheets, please use the sheet called: ***6. Precio OIC Mensual***

It will look like this:



Precios indicativos OIC por grupos - Promedio Mensual

Centavos de dólar por libra

Fuente: ICO

		Suaves colombianos (arábigo)			Otros suaves (arábigo)			Naturales del Brasil (arábigo)			Robustas		
Mes	Precio del indicador compuesto OIC	Nueva York	Europa	Promedio ponderado	Nueva York	Europa	Promedio ponderado	Nueva York	Europa	Promedio ponderado	Nueva York	Europa	Promedio ponderado
Jan-00	82.15	130.12	124.36	130.13	109.17	116.82	111.11	97.67	103.10	97.68	53.62	52.41	53.18
Feb-00	76.15	124.72	118.67	124.73	101.17	110.19	103.44	91.51	96.58	91.51	49.41	47.97	48.85
Mar-00	73.49	119.51	115.78	119.51	98.26	108.13	100.73	89.93	94.78	89.93	47.26	44.73	46.25
Apr-00	69.53	112.67	109.12	112.67	92.41	101.51	94.61	86.46	90.70	86.46	45.21	43.31	44.45
May-00	69.22	110.31	107.85	110.31	91.76	100.99	94.17	87.23	91.01	87.23	45.19	43.01	44.32
Jun-00	64.56	100.30	98.57	100.30	84.10	92.94	86.44	78.32	83.34	78.32	43.72	41.12	42.68

This data contains 4 sub-divisions of Coffee types:

Name in Spanish	**Name in English**
Suaves Colombianos (Arábigo)	Colombian Mild (Arabicas)
Otros Suaves (Arábigo)	Other Mild (Arabicas)
Naturales de Brazil (Arabigo)	Brazilian (Arabicas)
Robustas	Robustas

There are 14 Columns, the names have been translated or changed as per the following table.

Subdivision Coffee Type	Column Original name	Column New name
	Mes	Date
	Precio del indicador compuesto OIC	OIC_price
Suaves Colombianos (Arábigos)	Nueva York	Colombia_ny
	Europa	Colombia_europe
	Promedio ponderado	Colombia_average
Otros suaves (Arábigo)	Nueva York	Other_ny
	Europa	Other_europe
	Promedio ponderado	Other_average
Naturales de Brazil (Arabigo)	Nueva York	Brazil_ny
	Europa	Brazil_europe
	Promedio ponderado	Brazil_average
Robustas	Nueva York	Robustas_ny
	Europa	Robustas_europe
	Promedio ponderado	Robustas_average

Columns Metadata

Date: Column expressing the date monthly beginning January of 2000, all prices will have a reference for this date.

OIC_price: Is the average price of the International Coffee Organization for the month and year shown, measured in US cents/lb

Colombia_ny: Are the average price of Colombian Mild Arabicas for the month and year shown and is expressed in US cents/lb, in the US market

Colombia_europe: Is the average price of Colombian Mild Arabicas for the month and year shown and is expressed in US cents/lb, in Germany and France

Colombia_average: Is the weighted average Colombian Mild Arabicas Composite Indicator Price, as per "Section 4 "of the Indicator Prices SC-106/21 for the month and year shown and is expressed in US cents/lb

Other_ny: Is the average price of Other Mild Arabicas for the month and year shown and is expressed in US cents/lb, in the US market

Other_europe: Is the average price of Other Mild Arabicas for the month and year shown and is expressed in US cents/lb, in Germany and France

Other_average: Is the average price of Other Mild Arabicas Composite Indicator Price, as per "Section 4 "of the Indicator Prices SC-106/21 for the month and year shown and is expressed in US cents/lb

Brazil_ny: Is the average price of Brazilian Naturals for the month and year shown and is expressed in US cents/lb, in the US market

Brazil_europe: Is the average price of Brazilian Naturals for the month and year shown and is expressed in US cents/lb, in Germany and France

Brazil_average: Is the average price of Brazilian Naturals Composite Indicator Price, as per "Section 4 "of the Indicator Prices SC-106/21 for the month and year shown and is expressed in US cents/lb

Robustas_ny: Is the average price of Robustas for the month and year shown and is expressed in US cents/lb, in the US market

Robustas_europe: Is the average price of Robustas for the month and year shown and is expressed in US cents/lb, in Germany and France

Robustas_average: Is the average price of Robustas Composite Indicator Price, as per "Section 4 "of the Indicator Prices SC-106/21 for the month and year shown and is expressed in US cents/lb

Multivariate Analysis

Summary Statistics

There are 272 rows

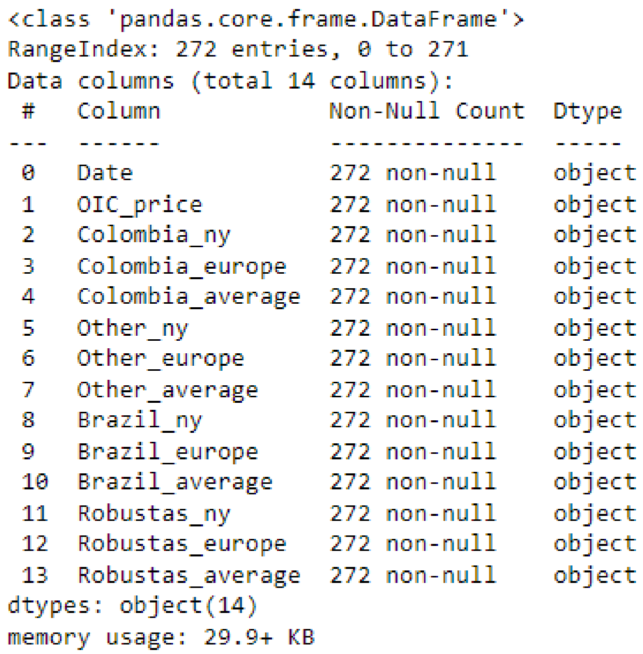
The data does not have NA's or empty cells

Measures of central tendency and measures of dispersion table:

	count	mean	std	min	25%	50%	75%	max
OIC_price	272	113.633237	43.687941	41.17	88.5475	113.155682	133.130252	231.24
Colombia_ny	272	153.333888	62.437782	58.92	112.94	144.413636	182.577237	319.63375
Colombia_europe	272	148.409523	59.732402	57.72	111.6975	141.123636	178.760455	311.45
Colombia_average	272	151.222932	61.156478	58.1	112.67	143.529552	179.0225	312.95
Other_ny	272	145.099616	58.41441	51.95	108.72	141.896818	169.232857	303.59
Other_europe	272	143.098234	55.916915	55.76	110.295	138.104348	165.942045	297.22
Other_average	272	143.96633	56.927377	54.28	109.7125	140.704773	166.686126	300.12
Brazil_ny	272	118.365112	49.732689	37.67	94.405	111.98	132.726023	271.39
Brazil_europe	272	123.408718	51.170812	38.71	96.056883	117.983409	143.199599	273.43
Brazil_average	272	121.959193	50.984818	38.63	95.605714	116.833333	140.72888	273.4
Robustas_ny	272	79.670588	28.448193	21.25	57.895	84.680554	103.530147	126.3
Robustas_europe	272	74.456376	26.5673	22.79	54.765	78.150682	97.329432	121.3
Robustas_average	272	75.376692	26.767165	22.81	55.3475	79.203636	98.367841	121.98

Note that Colombian_average maximum is almost 3 times the price of the Robustas_average which is predominantly from Vietnam, the reason is that "Robustas has a high caffeine content (2% to 4%), so the flavor is not as pure as Arabica" (Roldan Perez et al., 2009,32), "quality of the Robusta produced is uneven because of processing technology, drying equipment and post-harvest technological problems. These cause the coffee beans to have a high humidity level, and not meet the required standard of color, quality, and so on. This is the reason that Vietnam's coffee price is lower than the world price." (Roldan Perez et al., 2009,32).

The 3rd Quantil (75%) of Robustas_average is below the 1st Quantil (25%) of Colombian_average and Other_average, and almost the same for the 1st Quantil (25%) of Brazil_average.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 272 entries, 0 to 271
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Date              272 non-null    object
 1   OIC_price         272 non-null    object
 2   Colombia_ny       272 non-null    object
 3   Colombia_europe   272 non-null    object
 4   Colombia_average  272 non-null    object
 5   Other_ny          272 non-null    object
 6   Other_europe      272 non-null    object
 7   Other_average     272 non-null    object
 8   Brazil_ny         272 non-null    object
 9   Brazil_europe     272 non-null    object
 10  Brazil_average    272 non-null    object
 11  Robustas_ny       272 non-null    object
 12  Robustas_europe   272 non-null    object
 13  Robustas_average  272 non-null    object
dtypes: object(14)
memory usage: 29.9+ KB
```

There are a total of 14 Columns, each with 272 rows, and the type of data per column is the type Object that is like a String type, which will have to be converted to float type and time series for the column Date

Checking for duplicates on each column [duplicated index]:

Colombia_europe [20 - 40] In the years 2001-09 and 2003-05, it was exactly the same price coffee.

Brazil_ny[75 - 89] In the years 2006-04 and 2007-06, it was exactly the same price coffee.

Other_ny [158 - 159] In the years 2013-03 and 2013-04, it was exactly the same price as the coffee

Other_europe [29 - 41] In the years 2002-06 and 2003-06, it was exactly the same price as the coffee

Robustas_europe [118 - 119 - 130 - 183] On years 2010-11 and 2020-04, it was exactly the same price of the coffee

Robustas_ny [35 - 39] On years 2002-12 and 2003-04, it was exactly the same price of the coffee

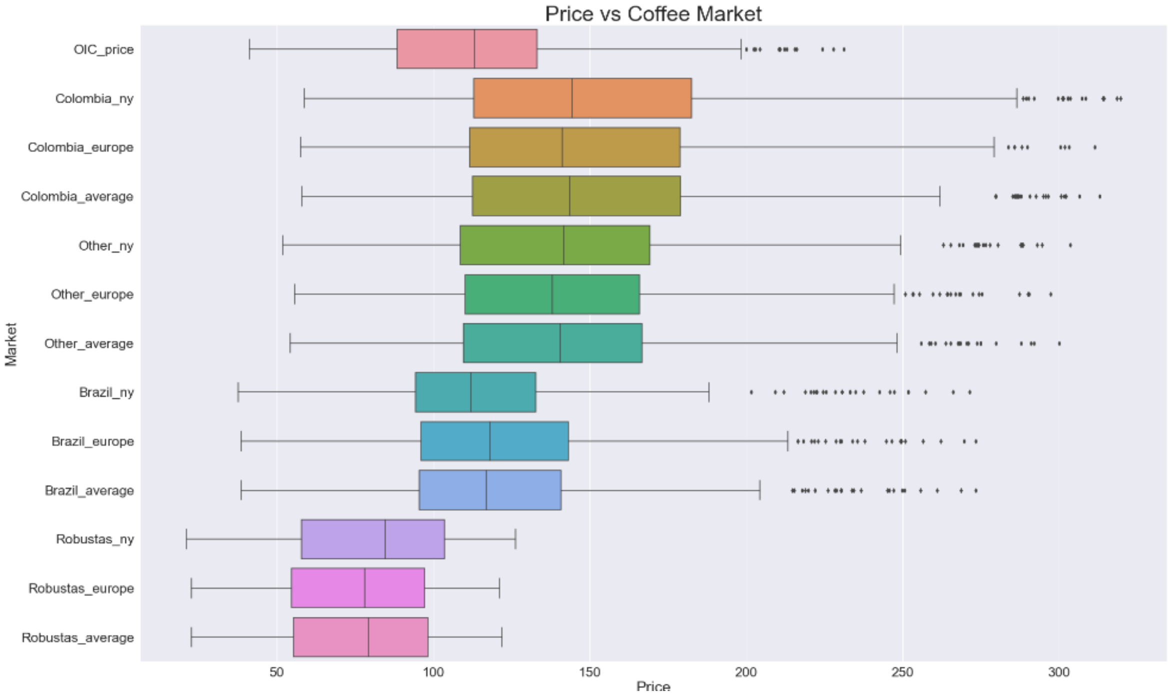
Min, Max, and Outliers

There are no outliers in the Robustas columns, but the other columns do have outliers

Column	Max Outlier	Min Outlier	# of Outliers
OIC_price	231.24	200	13
Colombia_ny	319.63	288.43	16
Colombia_europe	311.45	283.74	9
Colombia_average	312.95	279.55	19
Other_ny	303.59	262.94	19
Other_europe	297.22	250.75	19
Other_average	300.12	255.9	19
Brazil_ny	271.39	201.6	25
Brazil_europe	273.43	216.46	23
Brazil_average	273.4	214.8	24

Boxplot or candle sticks

Creating a boxplot for the columns, which can be seen the outliers graphically, we can appreciate that the variety of coffee Robustas, doesn't have outliers, but the rest of the varieties have outliers



The median of OIC_price is basically where the lower quartile Q1 of all "Colombian" columns start as well as the "Other" columns

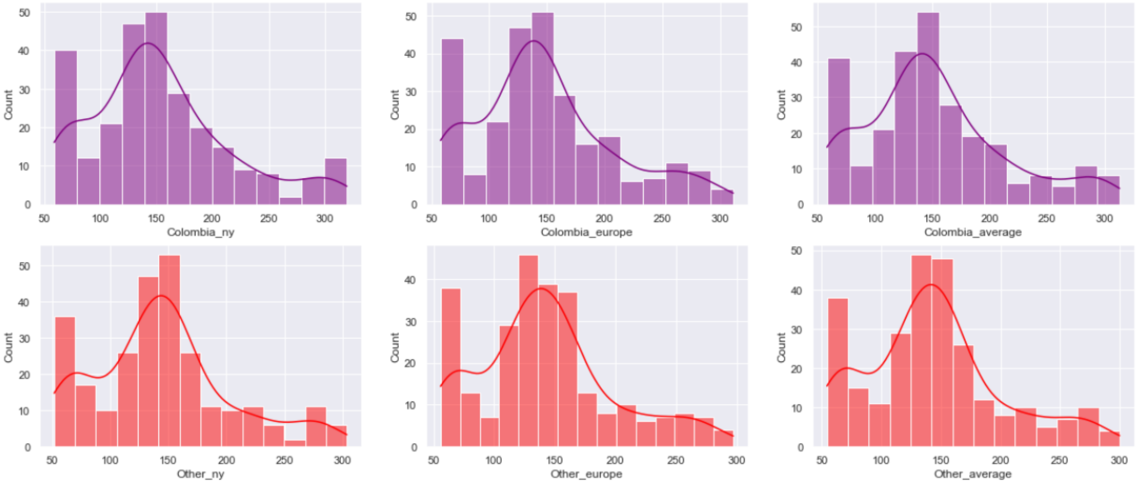
The median of "Robustas" columns is smaller than the lower quartile Q1 of OIC_price column

The upper quartile or Q3 of "Robustas" is smaller than the lower quartile Q1 of all the columns with the exception of OIC_price and "Robustas_ny"

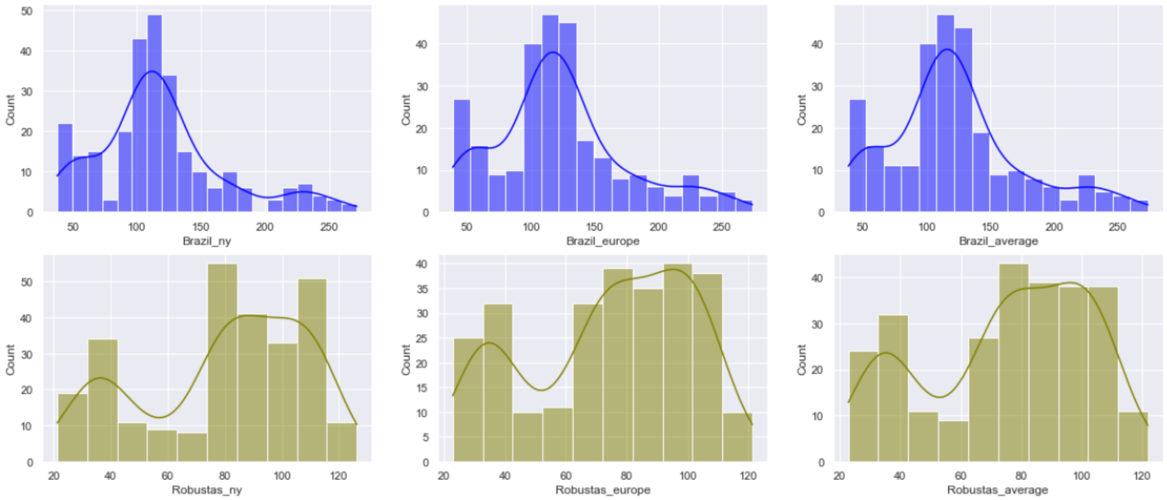
Brazil_ny has a small IQR, while all "Colombia" columns have a big IQR

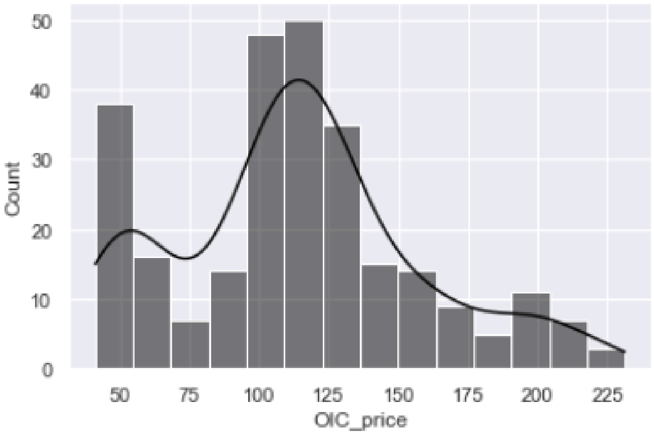
Frequency Distribution

Creating a histogram with each average variable:



All columns follow almost the same distribution pattern, however, Robustas_average is quite different from the others, and the data is not that "normally distributed". Please note that Robustas type coffee follows a different distribution that looks like a bimodal distribution, it has two local maxima points that are notable in the graph.



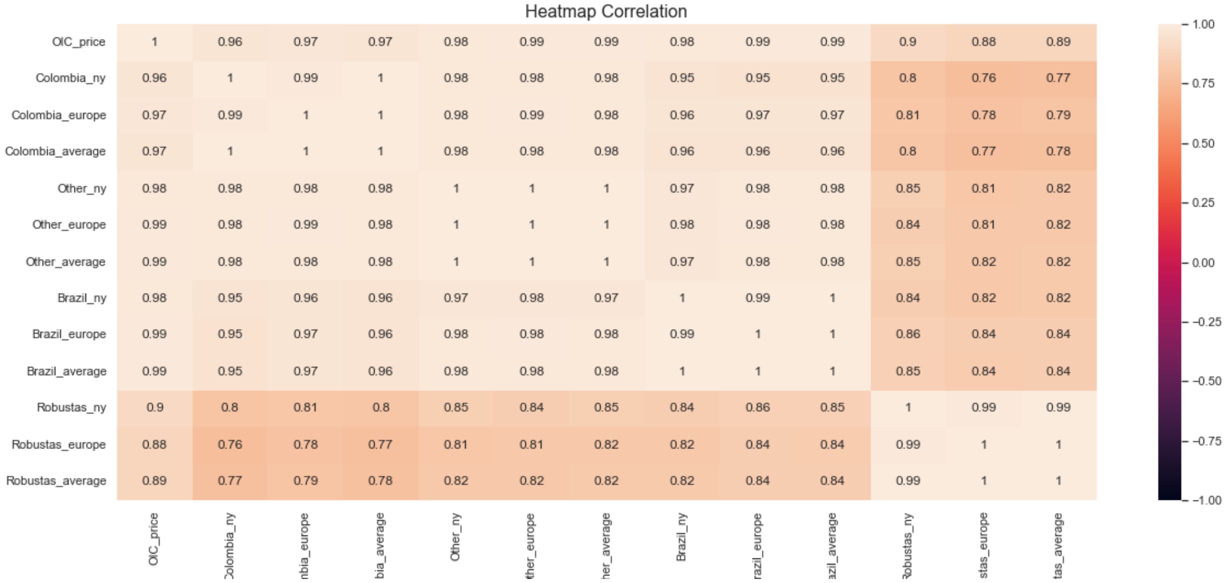


The variable OIC_price, here in color black, follows almost the same distribution of all the other columns, with a slight peak at around US $55 dollars.

Correlation and Heatmap

Creating a correlation matrix between all the columns, can be seen that there is a high correlation between all the averages, not that much with Robustas_average, but there is still a correlation, the lowest correlation is between Robustas_average and Colombia_ny with a value of 0.76, and with our target column which is OIC_price, the lowest correlation is Robustas_europe with 0.88

Overall, all columns have a high correlation between all of them, and it can be seen as well that the average columns have a correlation of 1 with the same group of coffee that they are averaging.

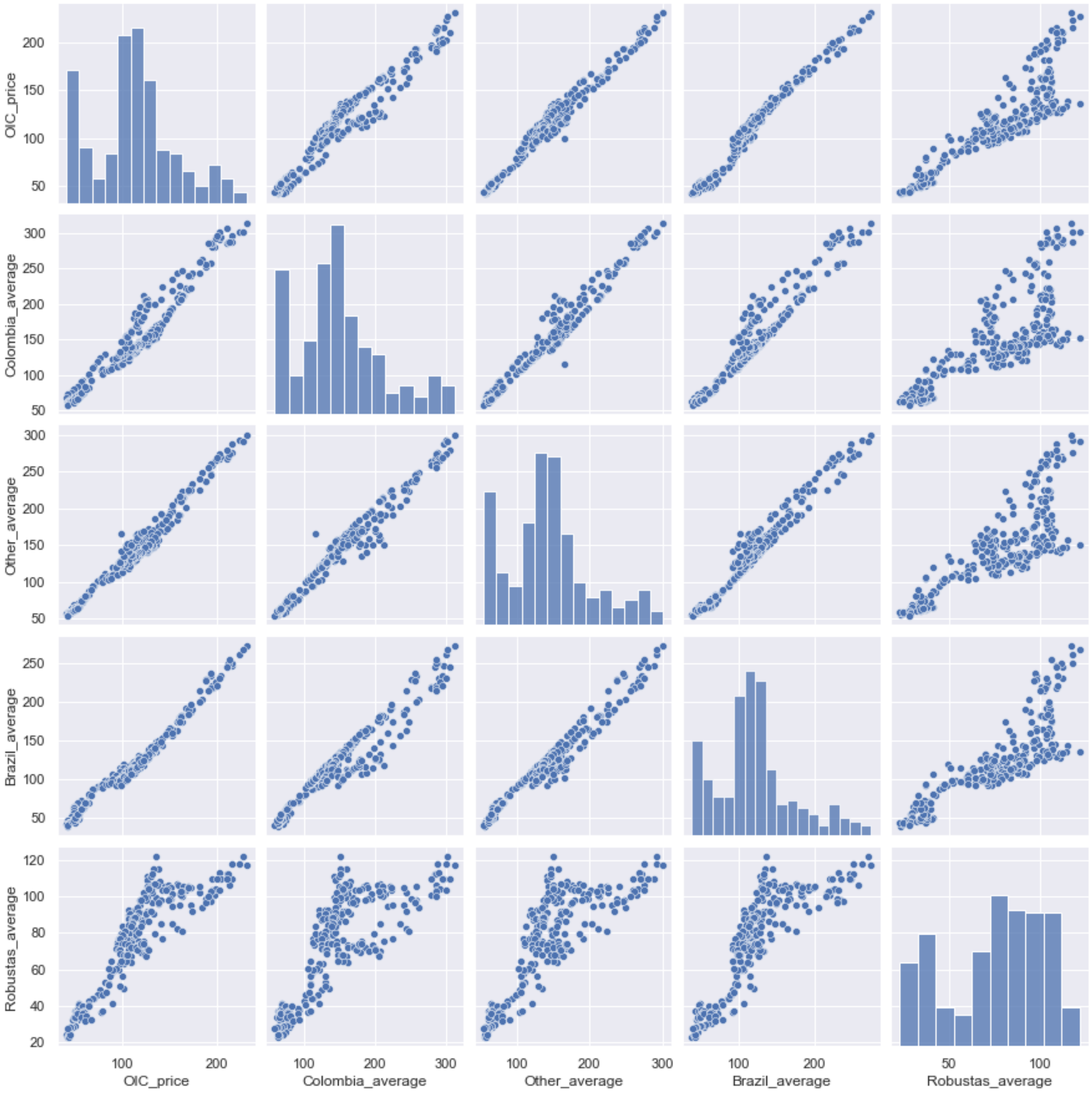


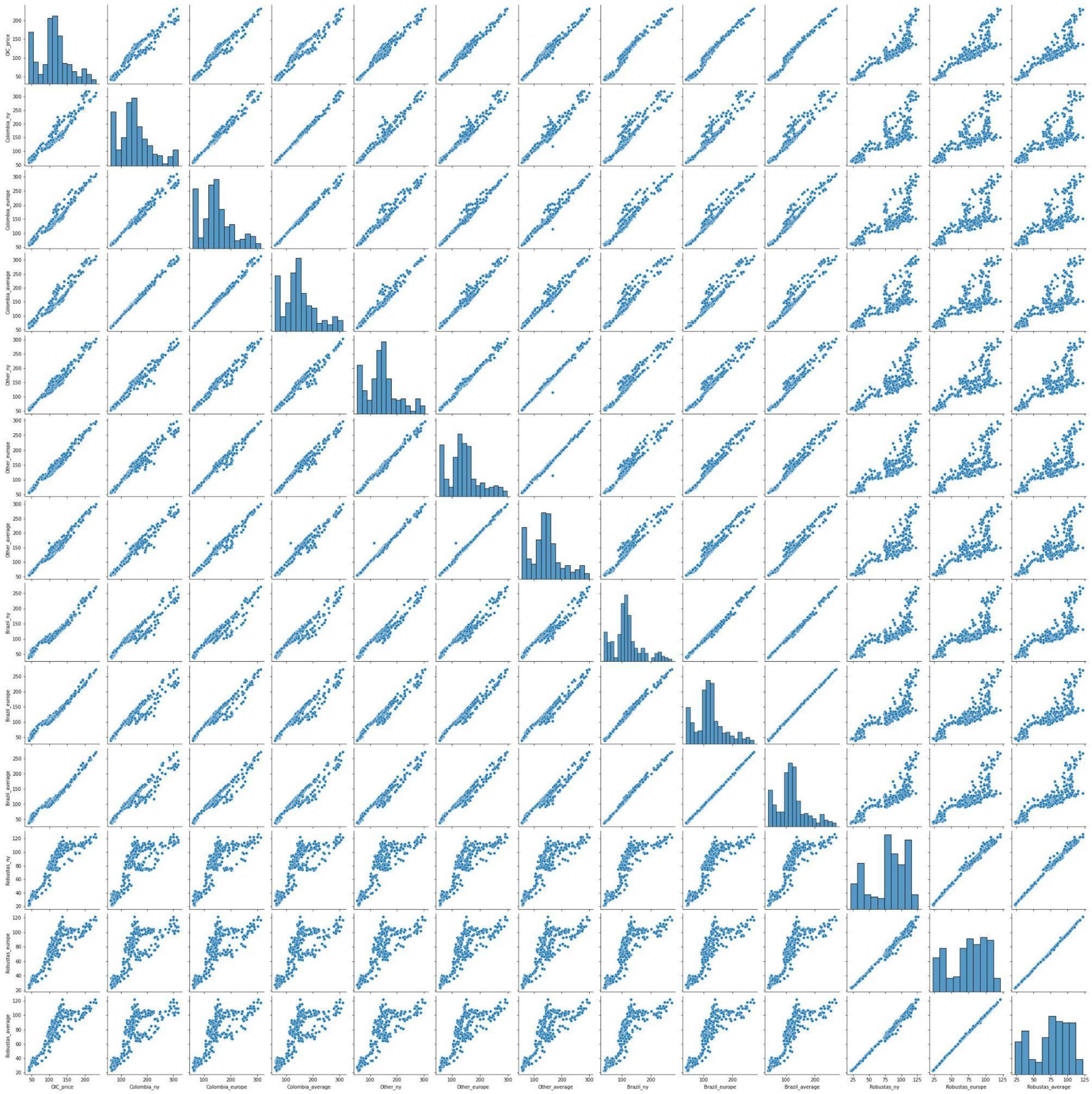
	OIC_price	Colombia_ny	nbia_europe	bia_average	Other_ny	ther_europe	her_average	Brazil_ny	razil_europe	azil_average	Robustas_ny	stas_europe	tas_average
OIC_price	1	0.96	0.97	0.97	0.98	0.99	0.99	0.98	0.99	0.99	0.9	0.88	0.89
Colombia_ny	0.96	1	0.99	1	0.98	0.98	0.98	0.95	0.95	0.95	0.8	0.76	0.77
Colombia_europe	0.97	0.99	1	1	0.98	0.99	0.98	0.96	0.97	0.97	0.81	0.78	0.79
Colombia_average	0.97	1	1	1	0.98	0.98	0.98	0.96	0.96	0.96	0.8	0.77	0.78
Other_ny	0.98	0.98	0.98	0.98	1	1	1	0.97	0.98	0.98	0.85	0.81	0.82
Other_europe	0.99	0.98	0.99	0.98	1	1	1	0.98	0.98	0.98	0.84	0.81	0.82
Other_average	0.99	0.98	0.98	0.98	1	1	1	0.97	0.98	0.98	0.85	0.82	0.82
Brazil_ny	0.98	0.95	0.96	0.96	0.97	0.98	0.97	1	0.99	1	0.84	0.82	0.82
Brazil_europe	0.99	0.95	0.97	0.96	0.98	0.98	0.98	0.99	1	1	0.86	0.84	0.84
Brazil_average	0.99	0.95	0.97	0.96	0.98	0.98	0.98	1	1	1	0.85	0.84	0.84
Robustas_ny	0.9	0.8	0.81	0.8	0.85	0.84	0.85	0.84	0.86	0.85	1	0.99	0.99
Robustas_europe	0.88	0.76	0.78	0.77	0.81	0.81	0.82	0.82	0.84	0.84	0.99	1	1
Robustas_average	0.89	0.77	0.79	0.78	0.82	0.82	0.82	0.82	0.84	0.84	0.99	1	1

Scatterplot Matrix

The scatterplot between all average columns and OIC_price confirms the correlations seen above on the matrix.

There is a linear relationship between the variables and the target, which is confirmed by the scatterplot.

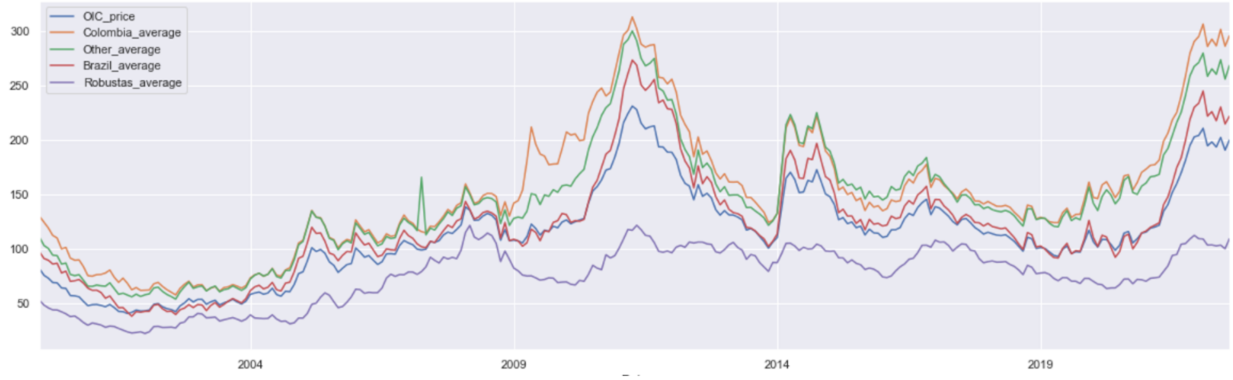




There is a linear relationship between almost all columns, but this can be seen better in a heat map.

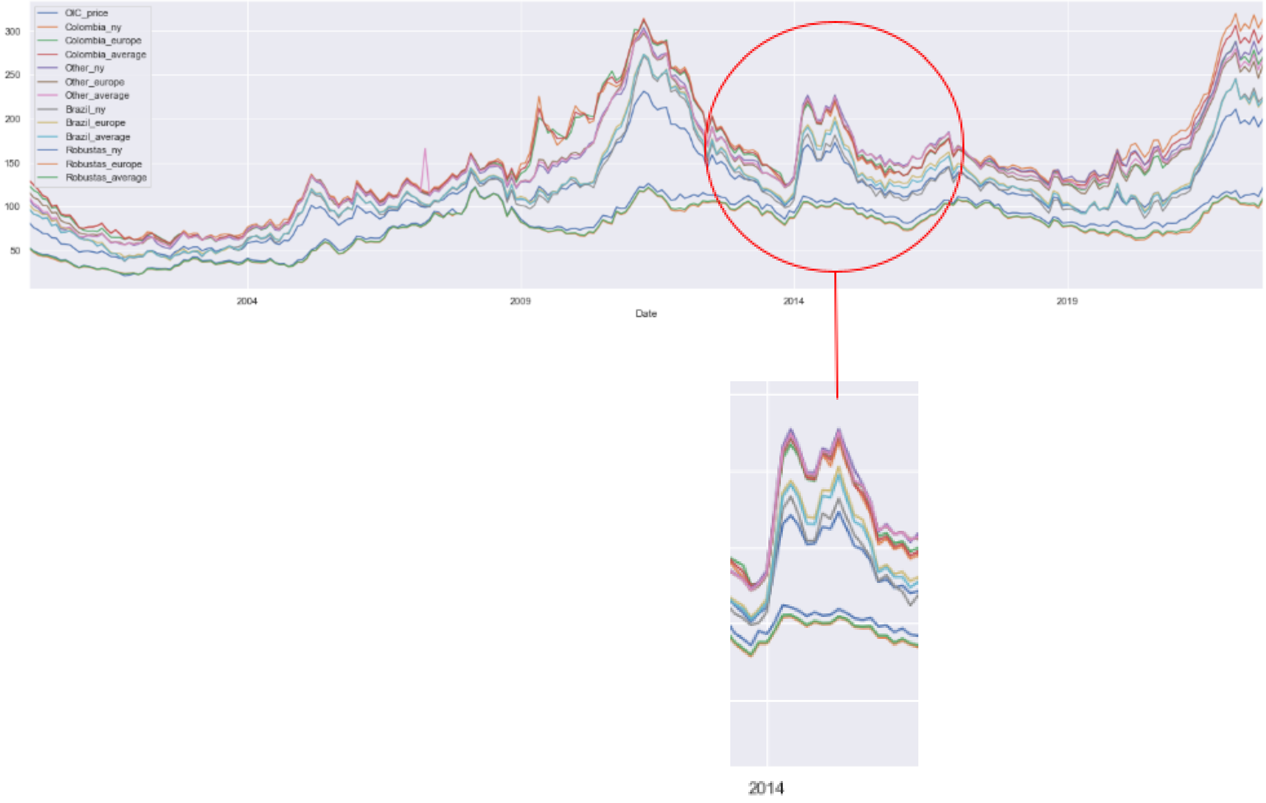
Exploratory Data Analysis

Coffee averages price distribution



The date column is on a monthly basis for a period of 10 years and behaves as a queue, where it has a front and rear end every time a new month is added to the end of the queue, the month on the front is withdrawn from the queue.

On the Graph distribution between the averages over time can be seen that there some coffees reached up to $300 dollars twice during the decade and had lots of peaks, while the robustas had a more flat-like behavior.

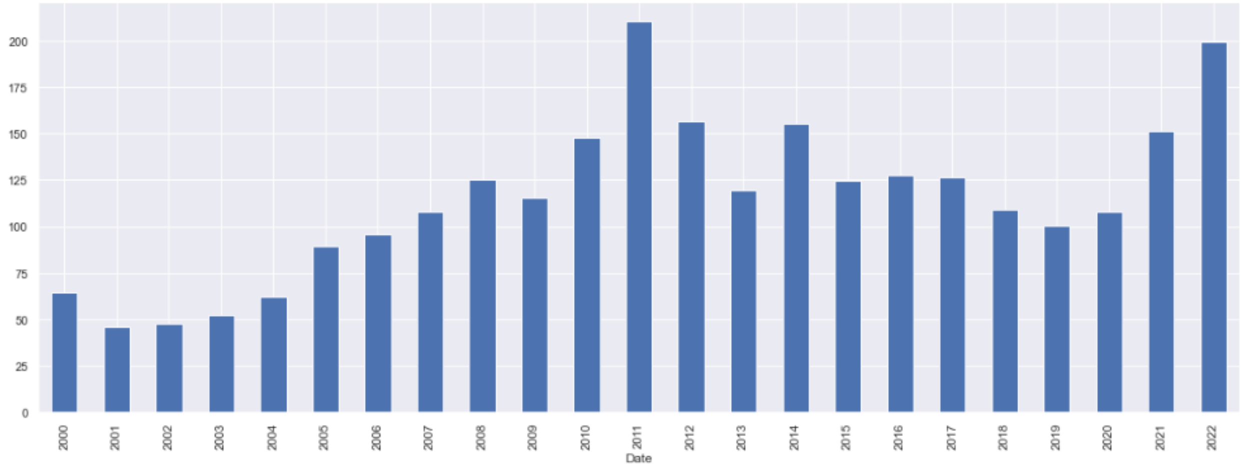


Note, that “the daily price increased by over 50% between 30 January and 10 March, as the ongoing drought in Brazil and uncertainty over the 2014/15 crop put upward pressure on prices” (ICO.ORG, 2014), this is particularly important due that January 2014 was one of the hottest months in “Brazil, which produces nearly 40% of the world’s coffee” (Wile, 2014), and lots of the crops were lost on this year

Average price per year

Date	OIC_ price	Colombia_ ny	Colombia_ europe	Colombia_ average	Other_ ny	Other_ europe	Other_ average	Brazil_ ny	Brazil_ europe	Brazil_ average	Robustas_ ny	Robustas_ europe	Robustas_ average
2000	64.25	102.60	99.80	102.60	85.09	92.89	87.08	79.86	83.67	79.86	42.12	40.36	41.41
2001	45.59	72.21	68.24	72.05	61.94	63.14	62.28	50.52	52.42	50.70	27.30	27.49	27.54
2002	47.76	65.27	64.78	64.90	60.44	62.35	61.55	45.10	45.92	45.23	30.84	29.78	30.03
2003	51.90	67.31	64.34	65.33	64.09	64.30	64.20	50.82	50.16	50.31	38.39	36.50	36.94
2004	62.15	83.85	79.49	81.44	80.15	80.64	80.47	68.18	69.11	68.97	37.28	35.66	35.98
2005	89.34	117.00	114.67	115.73	114.29	114.83	114.83	101.33	102.49	102.29	53.38	49.86	50.51
2006	95.75	117.92	115.70	116.80	113.95	114.80	114.40	102.88	104.19	103.92	70.28	66.98	67.56
2007	107.68	126.74	124.70	125.57	123.16	123.81	127.83	110.69	112.06	111.79	88.26	86.30	86.37
2008	125.47	146.08	144.27	145.37	139.63	141.99	141.21	124.47	129.48	128.35	107.66	106.32	106.56
2009	115.67	180.87	174.58	177.43	141.65	145.48	143.84	111.39	116.55	115.33	77.16	74.02	74.58
2010	148.16	224.62	227.08	226.33	195.44	197.62	196.97	146.68	156.92	154.66	85.07	78.46	79.55
2011	210.39	283.82	283.67	283.84	273.20	269.55	271.07	243.67	248.72	247.62	115.99	107.91	109.21
2012	156.36	203.95	200.53	202.15	187.59	185.76	186.53	171.37	176.13	175.03	110.58	101.30	102.76
2013	119.51	148.25	147.53	147.87	141.08	138.42	139.53	117.95	123.56	122.23	100.50	92.95	94.16
2014	155.26	198.09	198.16	197.95	202.85	199.08	200.39	161.30	175.29	171.59	105.60	99.47	100.43
2015	124.67	149.88	154.02	151.80	160.53	159.54	159.94	123.11	135.72	132.45	94.20	86.84	88.05
2016	127.38	155.58	155.37	155.38	164.63	163.49	163.88	124.18	142.72	137.86	94.28	87.47	88.63
2017	126.68	154.07	150.41	152.37	152.41	149.50	150.73	126.55	133.78	131.91	104.09	100.28	100.95
2018	109.04	139.59	133.26	136.70	137.40	129.10	132.73	109.62	115.10	113.65	88.34	84.03	84.80
2019	100.52	137.07	129.19	133.60	137.46	125.52	130.66	100.07	101.99	101.53	80.06	72.15	73.56
2020	107.94	166.28	146.10	157.67	156.81	146.77	150.72	102.46	107.86	106.42	78.20	66.68	68.75
2021	151.29	228.42	206.53	219.05	209.88	201.42	204.62	158.23	163.01	161.70	99.81	87.74	89.87
2022	199.43	309.18	272.31	293.69	277.86	260.35	266.66	228.77	225.65	226.49	114.73	103.73	105.49

Plotting the average price per year as a distribution graph, shows the Average Maximum Price of OIC has been US $210.38 in the year 2011, while the Average Minimum Price of IC has been US $ 45.59 in the year 2001.

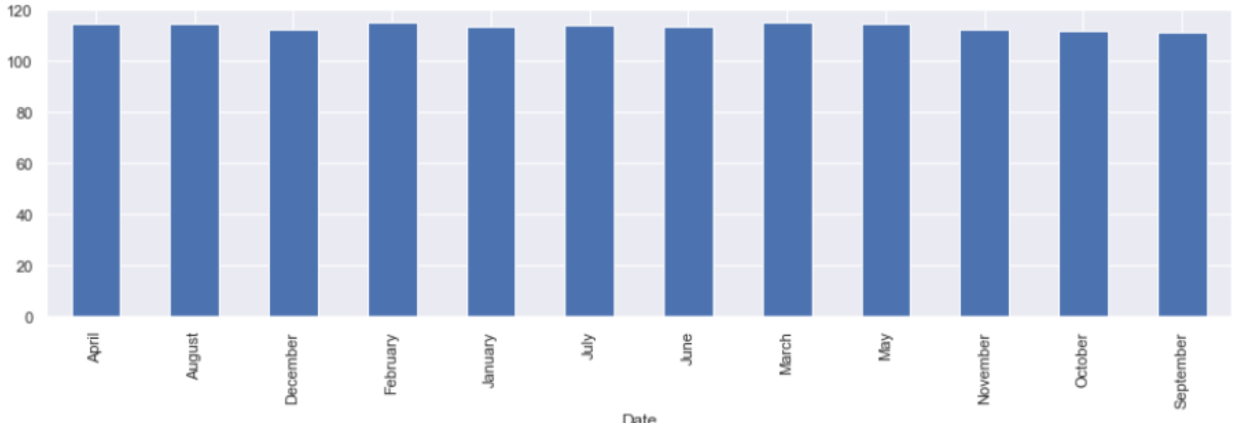


The months of February and March, are when the coffee has the highest prices, however, there is not a significant increase in price, only 3%

Minimum Month price: 111.48

Maximum Month price: 115.20

The graph shows the average price per month for OIC_price



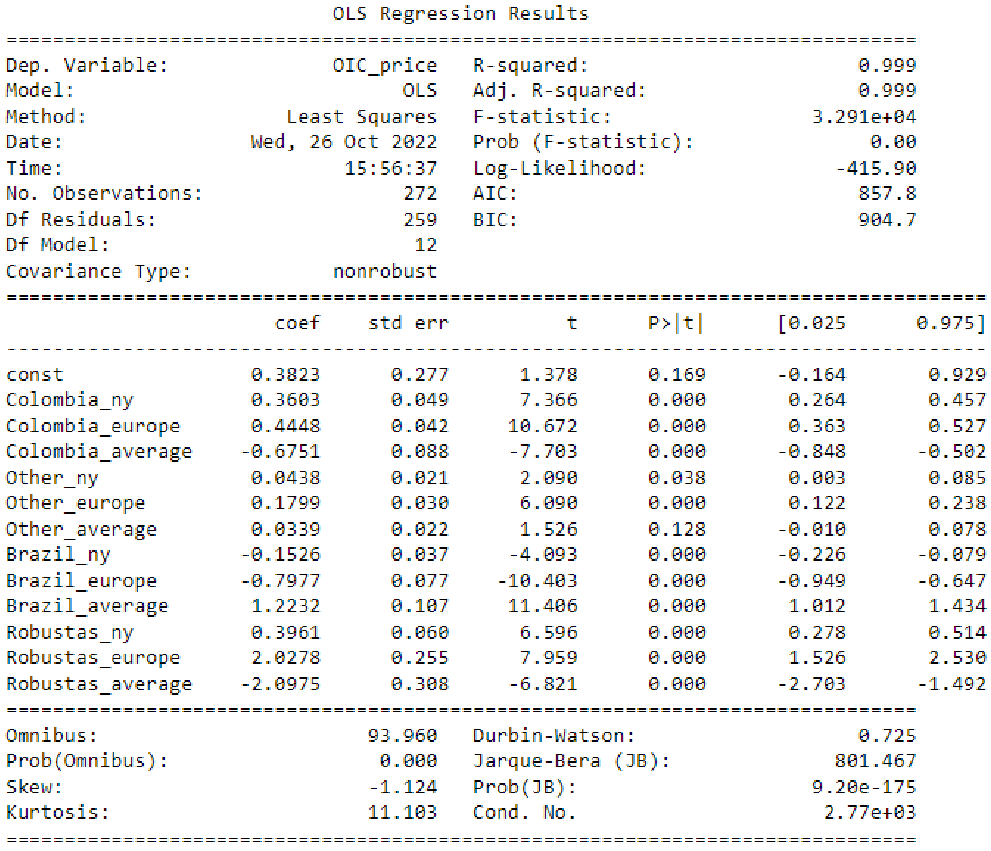
Features Selection and Backward Pairwise Relations

As there are 14 columns in our data set, in order to build a proper model to do regression, there will be a limit of SL = 0.05 and obtain the p-values for each of the columns, please note that the independent column is OIC_price, and there won't be an analysis for this column, and the same will be done with the column Date.

In order to obtain an equation that satisfies the model of multiple linear regression

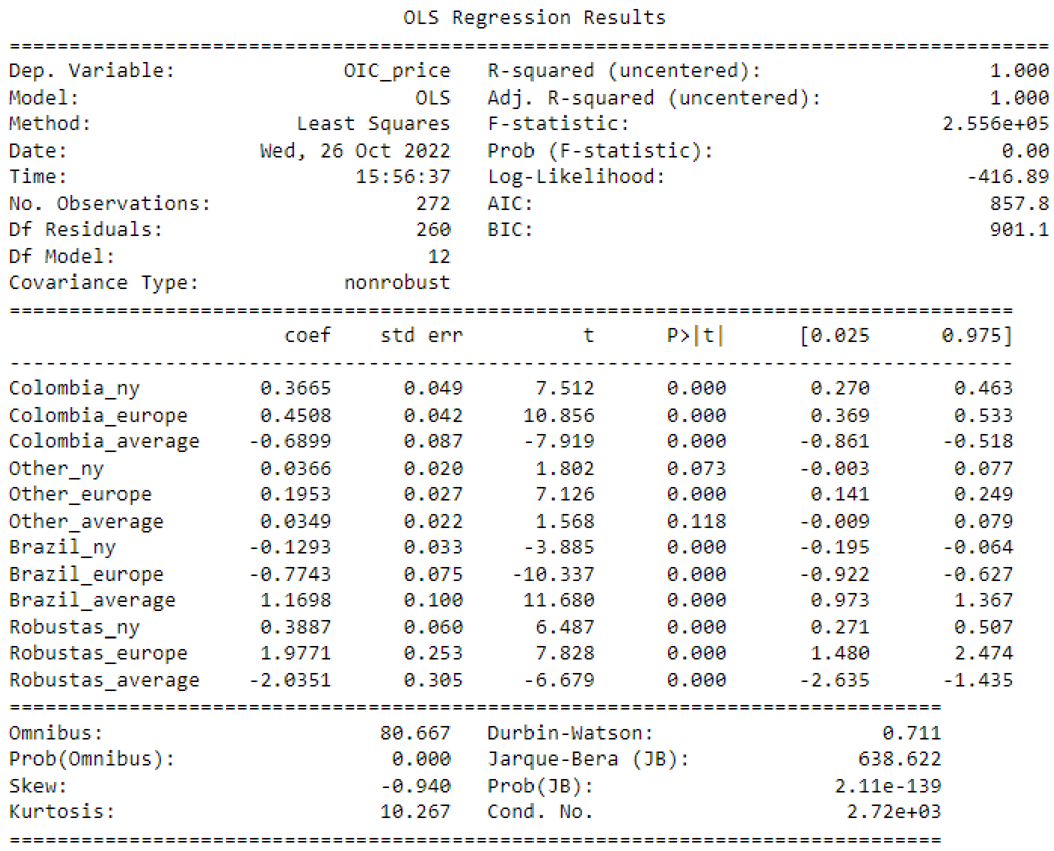
$$\hat{Y} = b_0 + b_1X_1 + b_2X_2 + \ldots + b_pX_p \,,$$

I'll obtain the p-value, using a backward stepwise regression, and if the p-values for the columns are > SL, then it will be discarded.



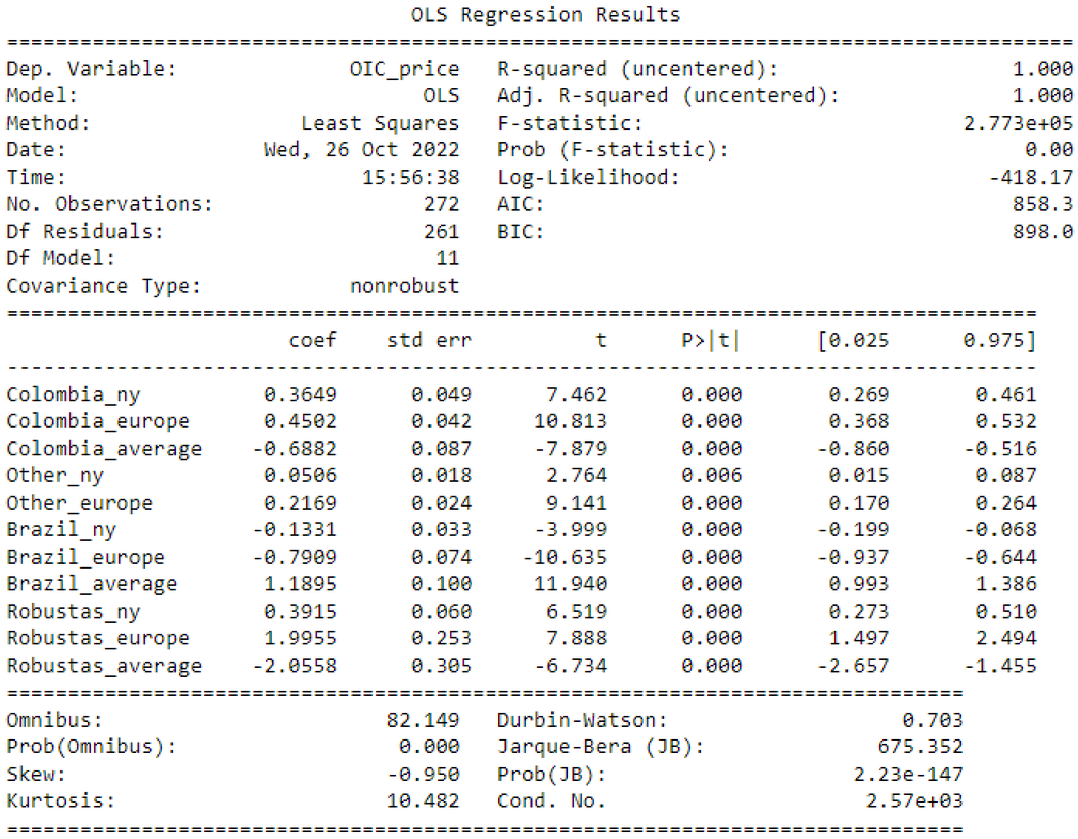
```
                            OLS Regression Results
==============================================================================
Dep. Variable:              OIC_price   R-squared:                       0.999
Model:                            OLS   Adj. R-squared:                  0.999
Method:                 Least Squares   F-statistic:                 3.291e+04
Date:                Wed, 26 Oct 2022   Prob (F-statistic):               0.00
Time:                        15:56:37   Log-Likelihood:                -415.90
No. Observations:                 272   AIC:                             857.8
Df Residuals:                     259   BIC:                             904.7
Df Model:                          12
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
const                0.3823      0.277      1.378      0.169      -0.164       0.929
Colombia_ny          0.3603      0.049      7.366      0.000       0.264       0.457
Colombia_europe      0.4448      0.042     10.672      0.000       0.363       0.527
Colombia_average    -0.6751      0.088     -7.703      0.000      -0.848      -0.502
Other_ny             0.0438      0.021      2.090      0.038       0.003       0.085
Other_europe         0.1799      0.030      6.090      0.000       0.122       0.238
Other_average        0.0339      0.022      1.526      0.128      -0.010       0.078
Brazil_ny           -0.1526      0.037     -4.093      0.000      -0.226      -0.079
Brazil_europe       -0.7977      0.077    -10.403      0.000      -0.949      -0.647
Brazil_average       1.2232      0.107     11.406      0.000       1.012       1.434
Robustas_ny          0.3961      0.060      6.596      0.000       0.278       0.514
Robustas_europe      2.0278      0.255      7.959      0.000       1.526       2.530
Robustas_average    -2.0975      0.308     -6.821      0.000      -2.703      -1.492
==============================================================================
Omnibus:                       93.960   Durbin-Watson:                   0.725
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              801.467
Skew:                          -1.124   Prob(JB):                    9.20e-175
Kurtosis:                      11.103   Cond. No.                     2.77e+03
==============================================================================
```

There are two columns with a high p-value, but only one will be removed in this step, in this case, the column with the constant value, which has a p-value of 0.169



```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:              OIC_price   R-squared (uncentered):                   1.000
Model:                            OLS   Adj. R-squared (uncentered):              1.000
Method:                 Least Squares   F-statistic:                          2.556e+05
Date:                Wed, 26 Oct 2022   Prob (F-statistic):                        0.00
Time:                        15:56:37   Log-Likelihood:                         -416.89
No. Observations:                 272   AIC:                                      857.8
Df Residuals:                     260   BIC:                                      901.1
Df Model:                          12
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
Colombia_ny          0.3665      0.049      7.512      0.000       0.270       0.463
Colombia_europe      0.4508      0.042     10.856      0.000       0.369       0.533
Colombia_average    -0.6899      0.087     -7.919      0.000      -0.861      -0.518
Other_ny             0.0366      0.020      1.802      0.073      -0.003       0.077
Other_europe         0.1953      0.027      7.126      0.000       0.141       0.249
Other_average        0.0349      0.022      1.568      0.118      -0.009       0.079
Brazil_ny           -0.1293      0.033     -3.885      0.000      -0.195      -0.064
Brazil_europe       -0.7743      0.075    -10.337      0.000      -0.922      -0.627
Brazil_average       1.1698      0.100     11.680      0.000       0.973       1.367
Robustas_ny          0.3887      0.060      6.487      0.000       0.271       0.507
Robustas_europe      1.9771      0.253      7.828      0.000       1.480       2.474
Robustas_average    -2.0351      0.305     -6.679      0.000      -2.635      -1.435
==============================================================================
Omnibus:                       80.667   Durbin-Watson:                   0.711
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              638.622
Skew:                          -0.940   Prob(JB):                    2.11e-139
Kurtosis:                      10.267   Cond. No.                     2.72e+03
==============================================================================
```

Then after removing the constant column, there are more columns again with different value, and once again, I'll remove the one with the highest p-value, in this case, is column number 6 called Other_average, with a p-value of 0.118

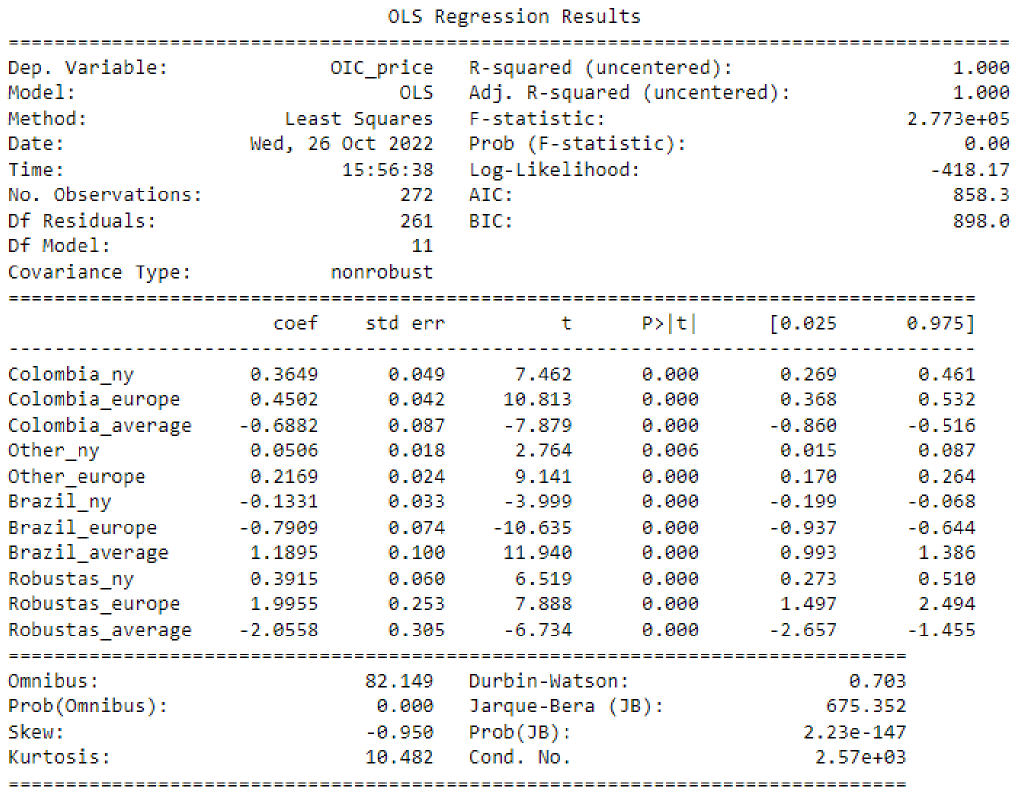


```
                                 OLS Regression Results
=======================================================================================
Dep. Variable:              OIC_price   R-squared (uncentered):                   1.000
Model:                            OLS   Adj. R-squared (uncentered):              1.000
Method:                 Least Squares   F-statistic:                          2.773e+05
Date:                Wed, 26 Oct 2022   Prob (F-statistic):                        0.00
Time:                        15:56:38   Log-Likelihood:                         -418.17
No. Observations:                 272   AIC:                                      858.3
Df Residuals:                     261   BIC:                                      898.0
Df Model:                          11
Covariance Type:            nonrobust
====================================================================================
                       coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
Colombia_ny          0.3649      0.049      7.462      0.000       0.269       0.461
Colombia_europe      0.4502      0.042     10.813      0.000       0.368       0.532
Colombia_average    -0.6882      0.087     -7.879      0.000      -0.860      -0.516
Other_ny             0.0506      0.018      2.764      0.006       0.015       0.087
Other_europe         0.2169      0.024      9.141      0.000       0.170       0.264
Brazil_ny           -0.1331      0.033     -3.999      0.000      -0.199      -0.068
Brazil_europe       -0.7909      0.074    -10.635      0.000      -0.937      -0.644
Brazil_average       1.1895      0.100     11.940      0.000       0.993       1.386
Robustas_ny          0.3915      0.060      6.519      0.000       0.273       0.510
Robustas_europe      1.9955      0.253      7.888      0.000       1.497       2.494
Robustas_average    -2.0558      0.305     -6.734      0.000      -2.657      -1.455
==============================================================================
Omnibus:                       82.149   Durbin-Watson:                   0.703
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              675.352
Skew:                          -0.950   Prob(JB):                    2.23e-147
Kurtosis:                      10.482   Cond. No.                     2.57e+03
==============================================================================
```

Now is the turn to remove the column Other_ny as it has a p-value = 0.06 and is above my limit of 0.05

The model is completed, keeping all the columns with the exemption of Columns: Other_average and Other_ny.

The coefficients b in the equation can be seen in the column **coef**, and the p-values can be seen in the column **p>|t|** and they all tend to zero, without being zero themselves.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:              OIC_price   R-squared (uncentered):          1.000
Model:                            OLS   Adj. R-squared (uncentered):     1.000
Method:                 Least Squares   F-statistic:                 2.773e+05
Date:                Wed, 26 Oct 2022   Prob (F-statistic):               0.00
Time:                        15:56:38   Log-Likelihood:                -418.17
No. Observations:                 272   AIC:                             858.3
Df Residuals:                     261   BIC:                             898.0
Df Model:                          11
Covariance Type:            nonrobust
===========================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------
Colombia_ny        0.3649      0.049      7.462      0.000       0.269       0.461
Colombia_europe    0.4502      0.042     10.813      0.000       0.368       0.532
Colombia_average  -0.6882      0.087     -7.879      0.000      -0.860      -0.516
Other_ny           0.0506      0.018      2.764      0.006       0.015       0.087
Other_europe       0.2169      0.024      9.141      0.000       0.170       0.264
Brazil_ny         -0.1331      0.033     -3.999      0.000      -0.199      -0.068
Brazil_europe     -0.7909      0.074    -10.635      0.000      -0.937      -0.644
Brazil_average     1.1895      0.100     11.940      0.000       0.993       1.386
Robustas_ny        0.3915      0.060      6.519      0.000       0.273       0.510
Robustas_europe    1.9955      0.253      7.888      0.000       1.497       2.494
Robustas_average  -2.0558      0.305     -6.734      0.000      -2.657      -1.455
==============================================================================
Omnibus:                       82.149   Durbin-Watson:                   0.703
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              675.352
Skew:                          -0.950   Prob(JB):                    2.23e-147
Kurtosis:                      10.482   Cond. No.                     2.57e+03
==============================================================================
```

The model:

Price_OIC = (0.3649***Colombia_ny**)+(0.4502***Colombia_europe**)-(0.6882***Colombia_average**)

+(0.0506***Other_ny**)+(0.2169***Other_europe**)-(0.1331***Brazil_ny**)-(0.7909***Brazil_europe**)

+(1.1895***Brazil_average**)+(0.3915***Robustas_ny**)+(1.9955***Robustas_europe**)

-(2.0558***Robustas_average**)

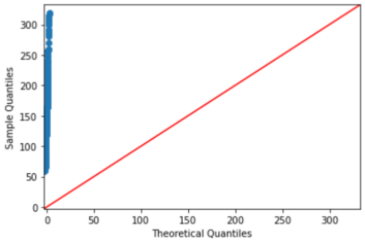
Normalization

This work will be done without Normalization and with Normalization, and at the end compare all the errors and values in a data frame, this is done for academic purposes and as a learning tool.

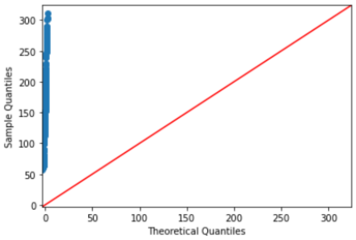
To normalize the data, I am going to create a data frame without the ['Date'] column, named **test**, then I'll select the **X** which contains the columns of the model, and **y** which contains the column with the independent variable "Price_OIC".

To confirm that normality has to be done, I'll use a Q-Q plot where it can be seen that the scatterplot doesn't follow the 45-degree line and needs to be normalized.

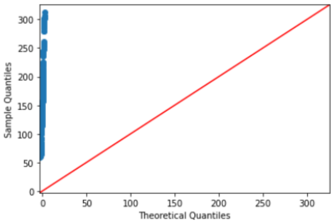
```
Column : Colombia_ny
```



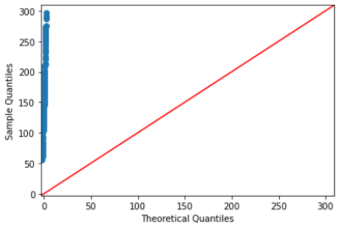
```
Column : Colombia_europe
```



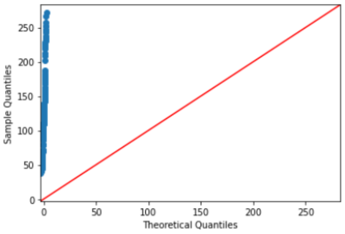
```
Column : Colombia_average
```



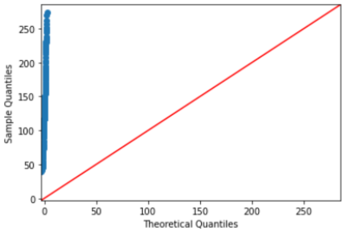
```
Column : Other_europe
```



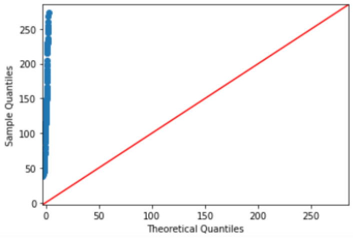
```
Column : Brazil_ny
```



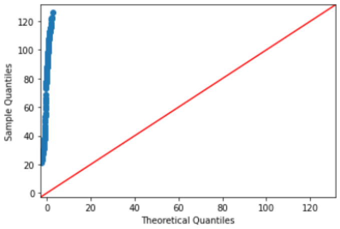
```
Column : Brazil_europe
```



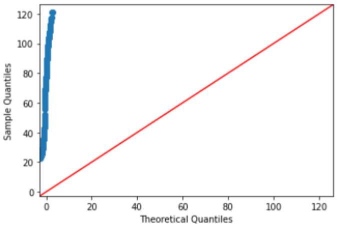
```
Column : Brazil_average
```



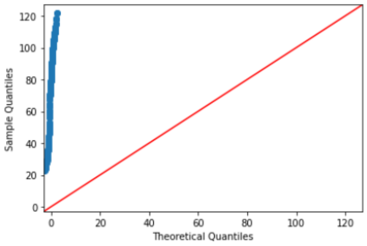
```
Column : Robustas_ny
```



```
Column : Robustas_europe
```



```
Column : Robustas_average
```

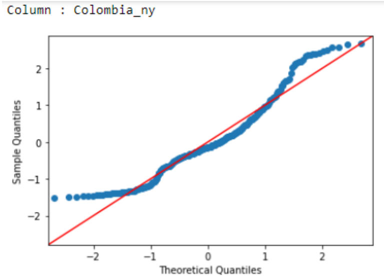


The function **StandardScaler()** will take care of the normalization:

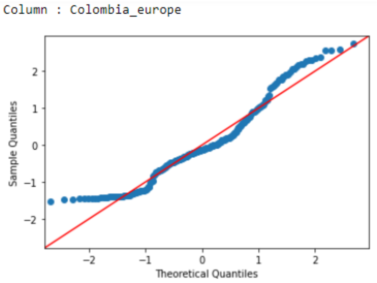
X = ['Colombia_ny', 'Colombia_europe', 'Colombia_average', 'Other_europe', 'Brazil_ny', 'Brazil_europe', 'Brazil_average', 'Robustas_ny', 'Robustas_europe', 'Robustas_average']

y = ['Price_OIC']

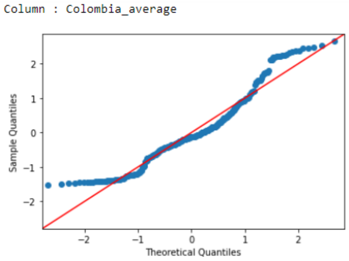
After normalizing, the same function to draw a Q-Q plot is applied to the columns and can be seen that now it follows normality.



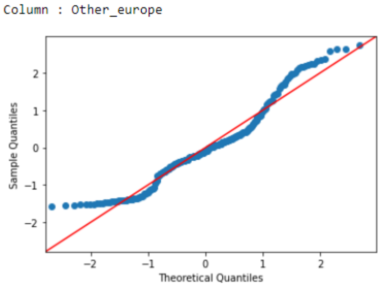
```
Column : Colombia_ny
```



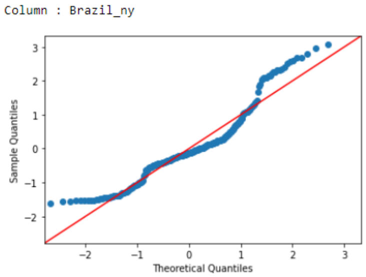
```
Column : Colombia_europe
```



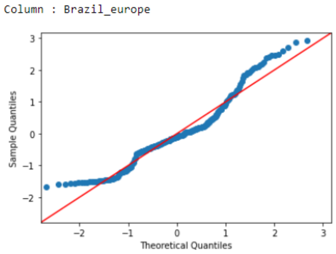
```
Column : Colombia_average
```



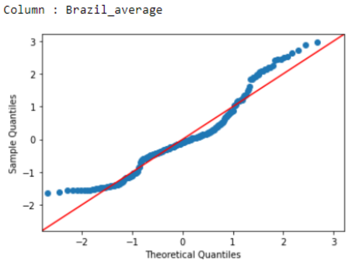
```
Column : Other_europe
```



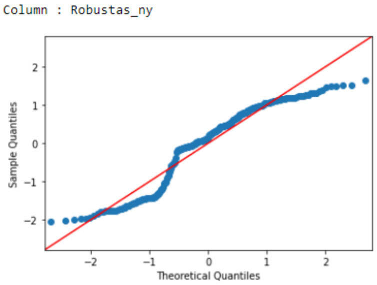
```
Column : Brazil_ny
```



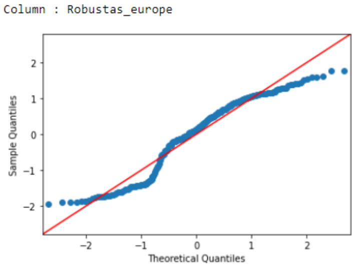
```
Column : Brazil_europe
```



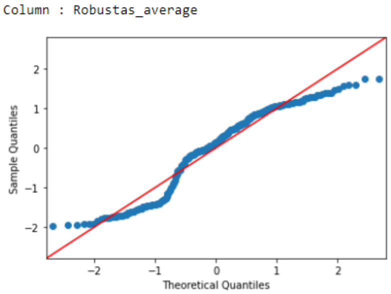
```
Column : Brazil_average
```



```
Column : Robustas_ny
```

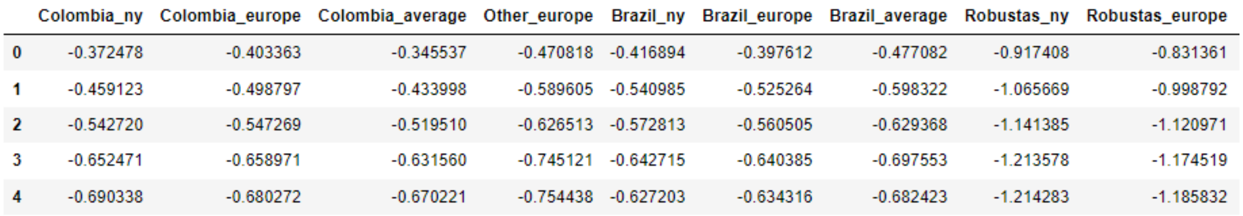


```
Column : Robustas_europe
```



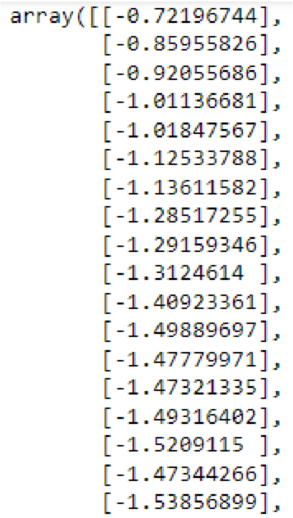
```
Column : Robustas_average
```

Screenshot of **X** normalized, not including all the columns as it won’t fit in the screenshot



	Colombia_ny	Colombia_europe	Colombia_average	Other_europe	Brazil_ny	Brazil_europe	Brazil_average	Robustas_ny	Robustas_europe
0	-0.372478	-0.403363	-0.345537	-0.470818	-0.416894	-0.397612	-0.477082	-0.917408	-0.831361
1	-0.459123	-0.498797	-0.433998	-0.589605	-0.540985	-0.525264	-0.598322	-1.065669	-0.998792
2	-0.542720	-0.547269	-0.519510	-0.626513	-0.572813	-0.560505	-0.629368	-1.141385	-1.120971
3	-0.652471	-0.658971	-0.631560	-0.745121	-0.642715	-0.640385	-0.697553	-1.213578	-1.174519
4	-0.690338	-0.680272	-0.670221	-0.754438	-0.627203	-0.634316	-0.682423	-1.214283	-1.185832

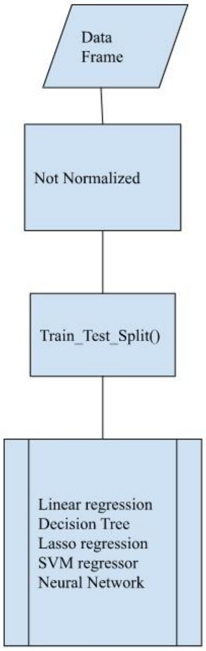
Screenshot of **y** Normalized, not including all the values as it won’t fit in the screenshot



```
array([[-0.72196744],
       [-0.85955826],
       [-0.92055686],
       [-1.01136681],
       [-1.01847567],
       [-1.12533788],
       [-1.13611582],
       [-1.28517255],
       [-1.29159346],
       [-1.3124614 ],
       [-1.40923361],
       [-1.49889697],
       [-1.47779971],
       [-1.47321335],
       [-1.49316402],
       [-1.5209115 ],
       [-1.47344266],
       [-1.53856899],
```

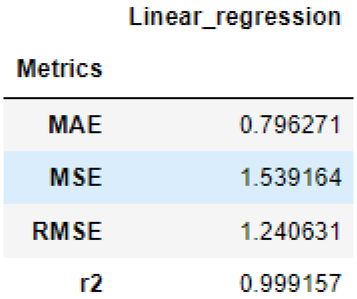
Not Normalized on Training and Testing Set

The following approach will be followed in this section



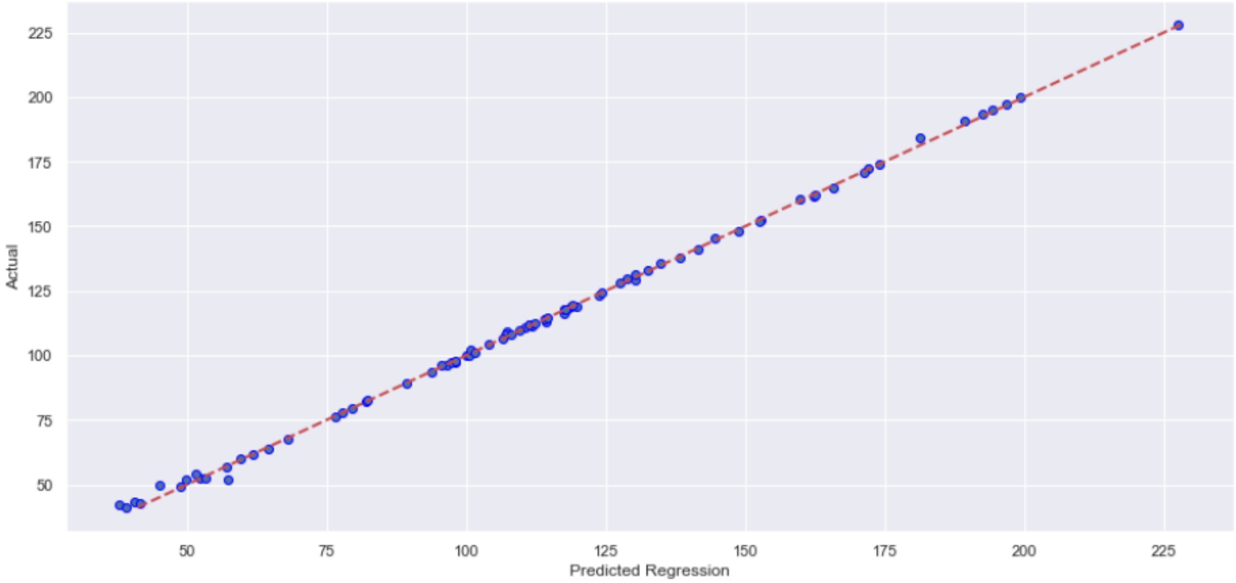
Linear Regression Not Normalized on Training and Testing Set

After creating the training and testing sets with the function train_test_split(), and dividing the sets 30% for testing and 70% for training, then I proceed to do a linear regression with the function LinearRegression(). Having the following metrics

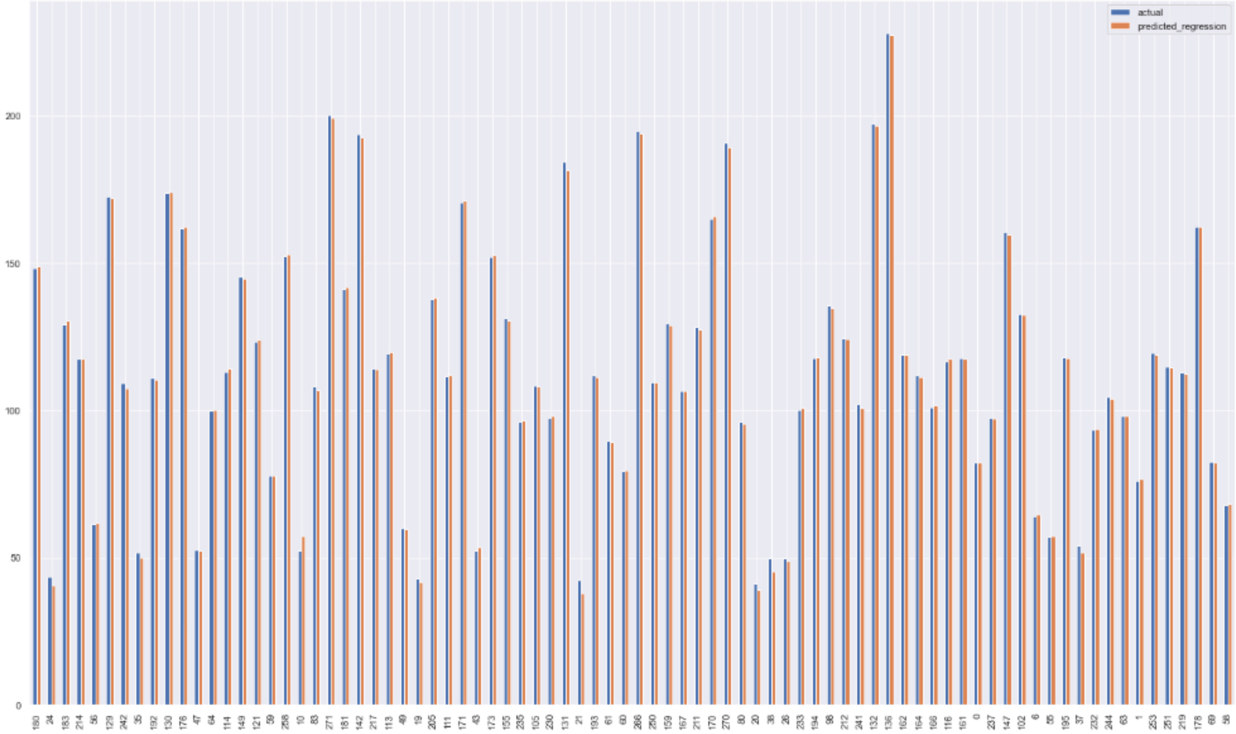


	Linear_regression
Metrics	
MAE	0.796271
MSE	1.539164
RMSE	1.240631
r2	0.999157

The results of the metrics and the plot of Actual vs predicted results, shows that the model fits properly.

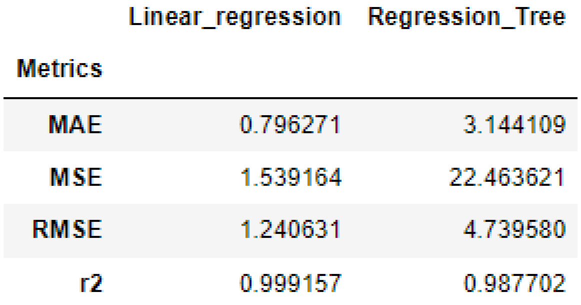


Just to see graphically how are the predictions vs the actual values behaving



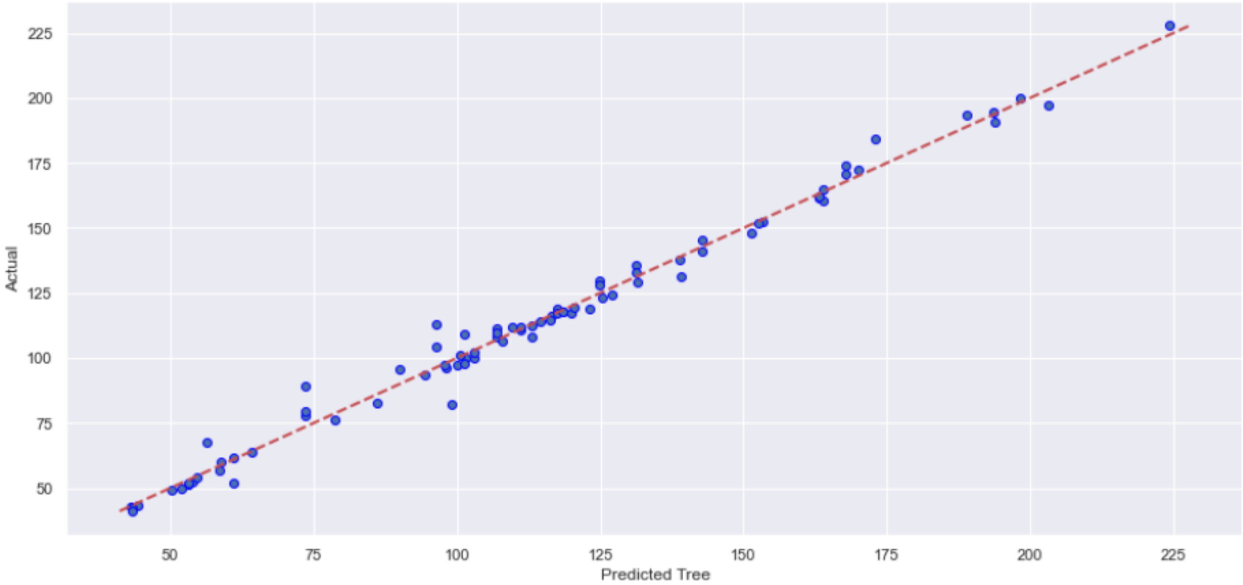
Decision Tree Regressor Not Normalized on Training and Testing Set

Another Algorithm that has been studied is the Decision Tree Regressor, here we have the following metrics and they are being compared with the previous algorithm



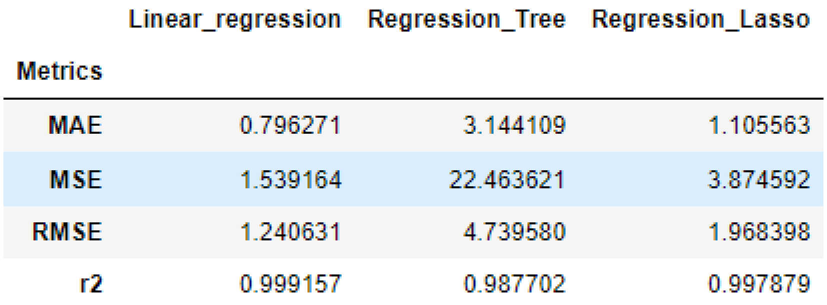
Metrics	Linear_regression	Regression_Tree
MAE	0.796271	3.144109
MSE	1.539164	22.463621
RMSE	1.240631	4.739580
r2	0.999157	0.987702

Can be seen that the metrics are worse than the Linear regression, having a RMSE very high when compared to the other one. The graph shows what we are talking about as we can see that some of the points are not that much following the line of the graph.



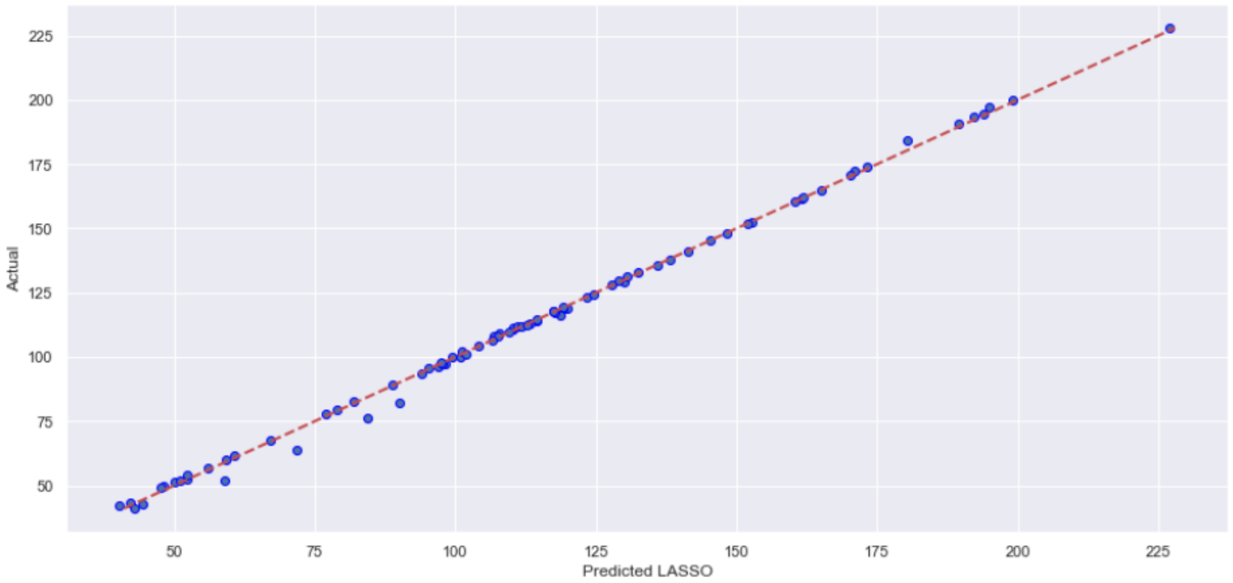
LASSO Regressor (Least Absolute Shrinkage and SelectionOperator) Not Normalized on Training and Testing Set

Another algorithm to evaluate our model is LASSO, and after evaluation of the model, the following is given and compared with the previous ones. When plot



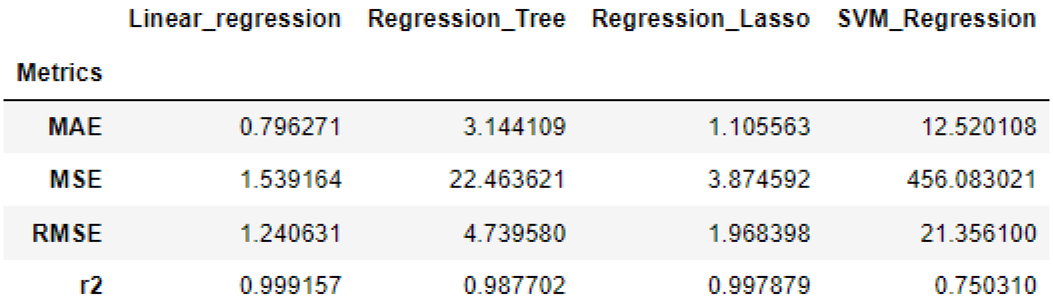
Metrics	Linear_regression	Regression_Tree	Regression_Lasso
MAE	0.796271	3.144109	1.105563
MSE	1.539164	22.463621	3.874592
RMSE	1.240631	4.739580	1.968398
r2	0.999157	0.987702	0.997879

When plotting the graph can be seen graphically that fits better than the Decision tree and this is confirmed by the metrics above having a better RMSE than the Decision tree but being still worse than the Linear Regression.



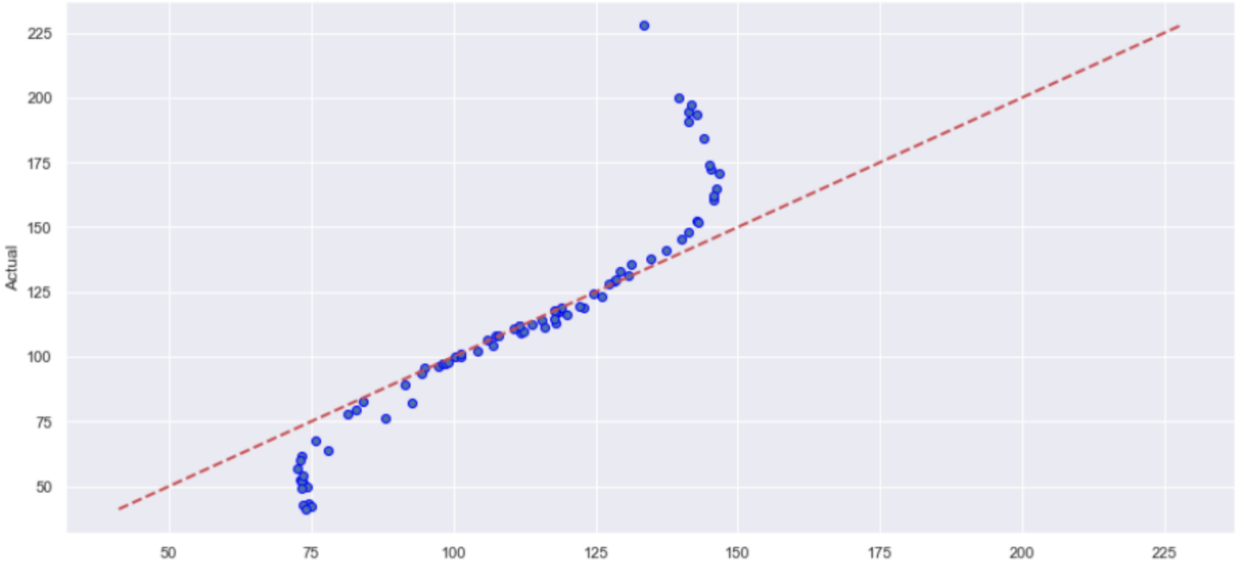
SVM (Support Vector Machine) Regressor Not Normalized on Training and Testing Set

The last of the algorithms being evaluated, looking at the metrics can be seen that has an awful MSE and therefore give us a very bad RMSE.



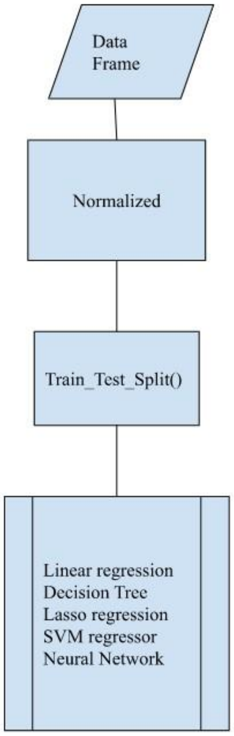
Metrics	Linear_regression	Regression_Tree	Regression_Lasso	SVM_Regression
MAE	0.796271	3.144109	1.105563	12.520108
MSE	1.539164	22.463621	3.874592	456.083021
RMSE	1.240631	4.739580	1.968398	21.356100
r2	0.999157	0.987702	0.997879	0.750310

Even the graph confirms that there is something wrong with the model as it is not fitting properly, SVM's are sensitive to feature scales and must be normalized when used, and it will be done further ahead.



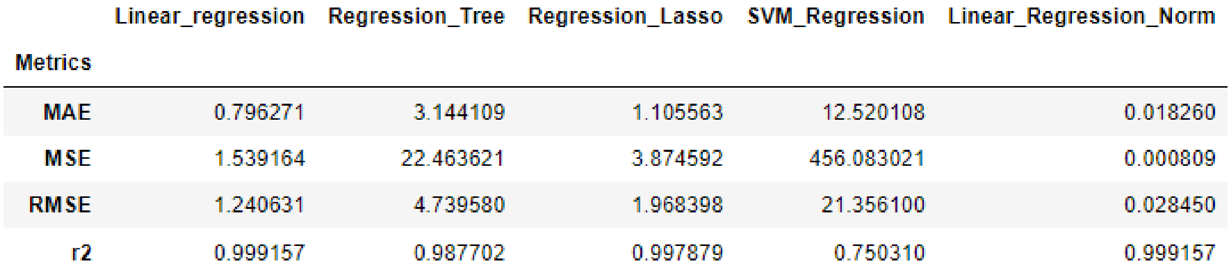
Normalized on Training and Testing Set

The following approach will be followed in this section



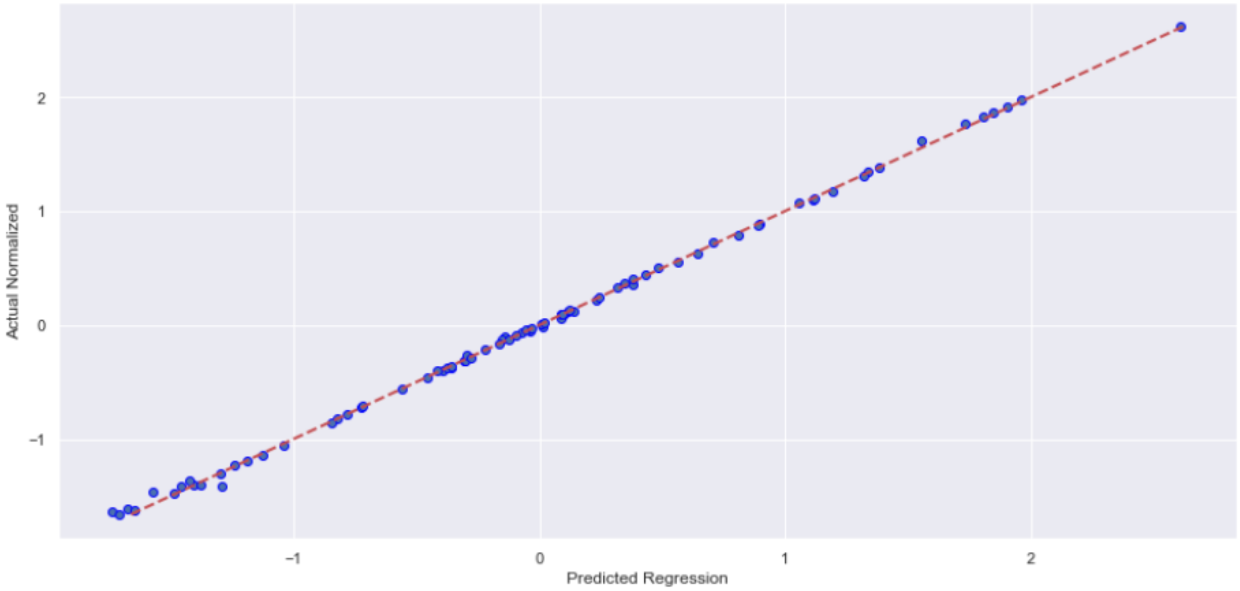
Linear Regression Normalized on Training and Testing Set

Now let’s apply the same algorithms to the model but this time normalizing as it was done at the Normalization stage. The same data has been split in 30% for testing and 70% for training, the models give us the following metrics and are still compared with the previous algorithms already seen.



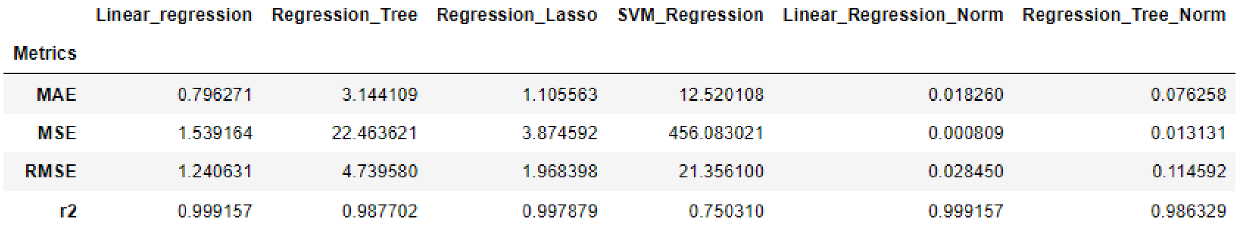
	Linear_regression	Regression_Tree	Regression_Lasso	SVM_Regression	Linear_Regression_Norm
Metrics					
MAE	0.796271	3.144109	1.105563	12.520108	0.018260
MSE	1.539164	22.463621	3.874592	456.083021	0.000809
RMSE	1.240631	4.739580	1.968398	21.356100	0.028450
r2	0.999157	0.987702	0.997879	0.750310	0.999157

Note that the metrics tend towards zero, meaning a much better performance than the algorithms not normalized and is confirmed graphically



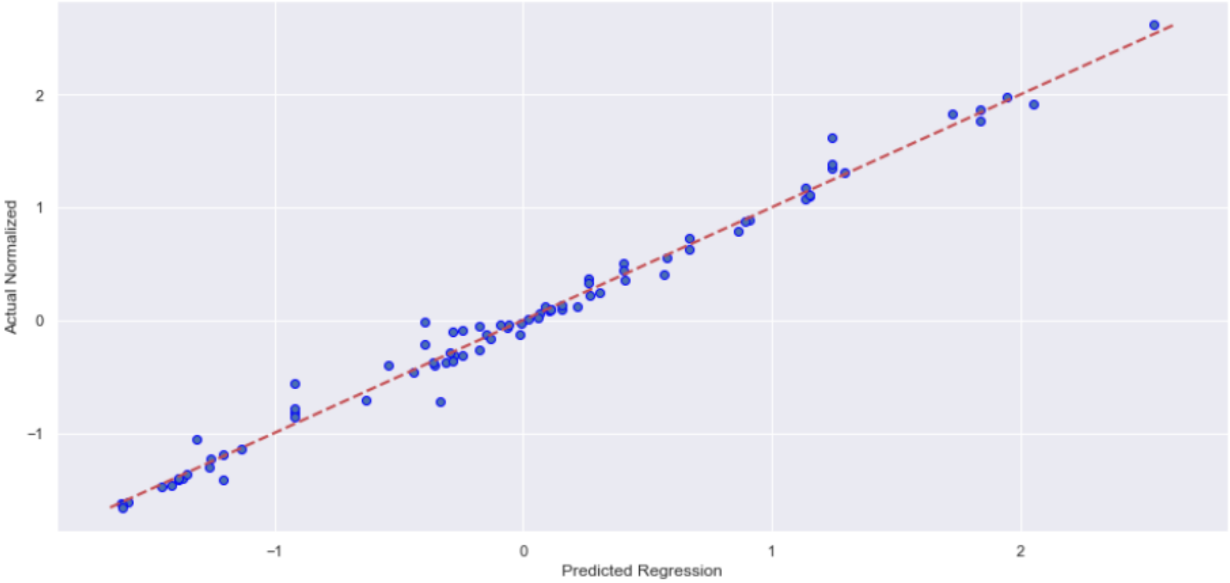
Decision Tree Regressor Normalized on Training and testing Set

The decision Tree regressor is analyzed in the same way as before but it has been already normalized, here we have the metrics



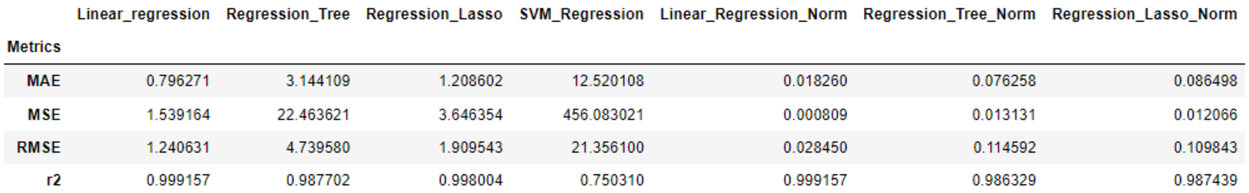
	Linear_regression	Regression_Tree	Regression_Lasso	SVM_Regression	Linear_Regression_Norm	Regression_Tree_Norm
Metrics						
MAE	0.796271	3.144109	1.105563	12.520108	0.018260	0.076258
MSE	1.539164	22.463621	3.874592	456.083021	0.000809	0.013131
RMSE	1.240631	4.739580	1.968398	21.356100	0.028450	0.114592
r2	0.999157	0.987702	0.997879	0.750310	0.999157	0.986329

Is a good predictor when compared with the Not Normalized one, but is still not that good when seen comparing the RMSE of linear regression Normalized, and can be seen on the graph.



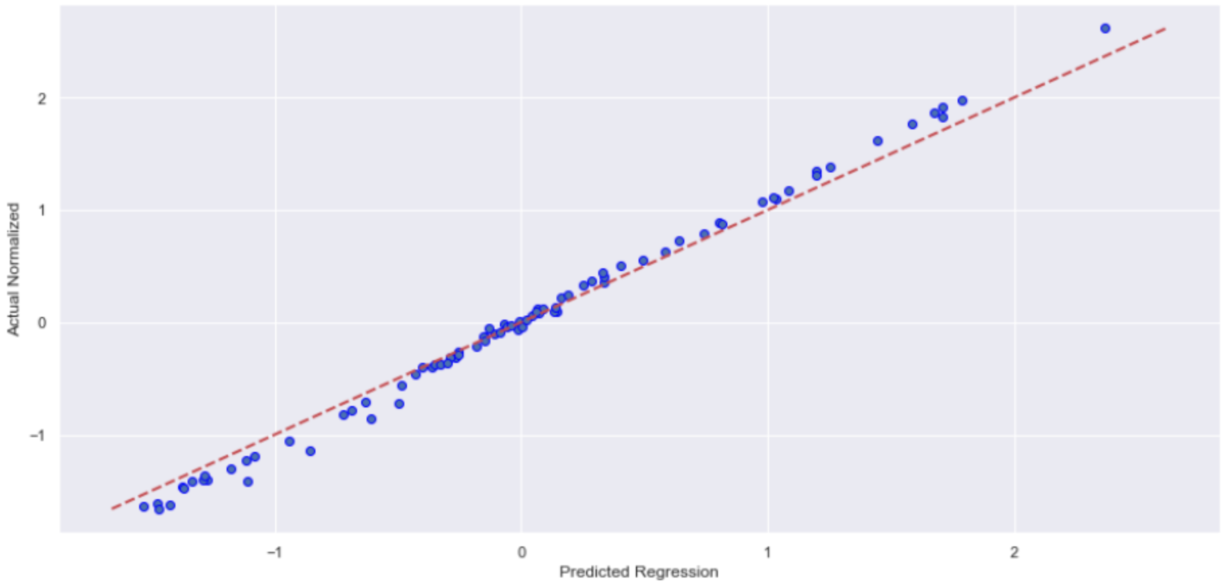
LASSO Regressor (Least Absolute Shrinkage and SelectionOperator) Normalized on Training and Testing Set

The model Lasso once it has been normalized produces the following metrics



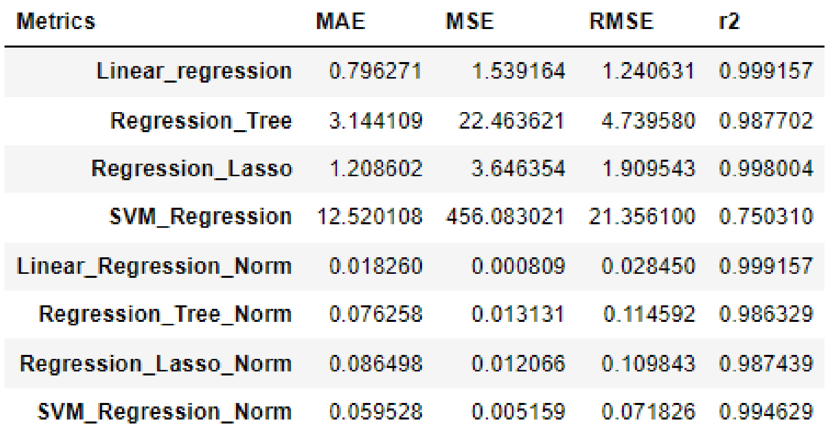
	Linear_regression	Regression_Tree	Regression_Lasso	SVM_Regression	Linear_Regression_Norm	Regression_Tree_Norm	Regression_Lasso_Norm
Metrics							
MAE	0.796271	3.144109	1.208602	12.520108	0.018260	0.076258	0.086498
MSE	1.539164	22.463621	3.646354	456.083021	0.000809	0.013131	0.012066
RMSE	1.240631	4.739580	1.909543	21.356100	0.028450	0.114592	0.109843
r2	0.999157	0.987702	0.998004	0.750310	0.999157	0.986329	0.987439

being this metrics much better than the Lasso Not Normalized, as the RMSE has been decreased by a lot and then I proceed with the graph



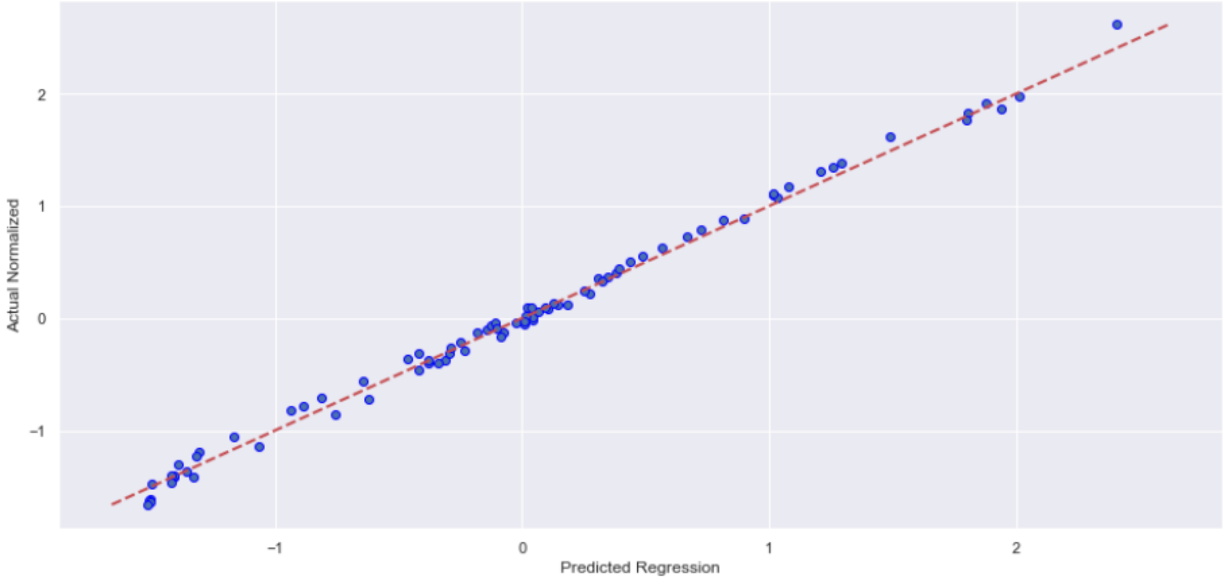
SVM (Support Vector Machine) Regressor Normalized on Training and Testing Set

The last of the algorithms being evaluated, transposing the metrics for aesthetics purposes



Metrics	MAE	MSE	RMSE	r2
Linear_regression	0.796271	1.539164	1.240631	0.999157
Regression_Tree	3.144109	22.463621	4.739580	0.987702
Regression_Lasso	1.208602	3.646354	1.909543	0.998004
SVM_Regression	12.520108	456.083021	21.356100	0.750310
Linear_Regression_Norm	0.018260	0.000809	0.028450	0.999157
Regression_Tree_Norm	0.076258	0.013131	0.114592	0.986329
Regression_Lasso_Norm	0.086498	0.012066	0.109843	0.987439
SVM_Regression_Norm	0.059528	0.005159	0.071826	0.994629

Can be seen as quite a change; it went from 21.36 to 0.07 in the RMSE metric, after being normalized, and now on the graph can be seen such a change.



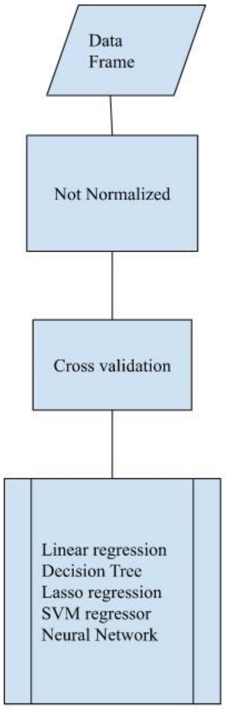
Not Normalized on Cross-Validation Set

The following approach will be followed in this section.

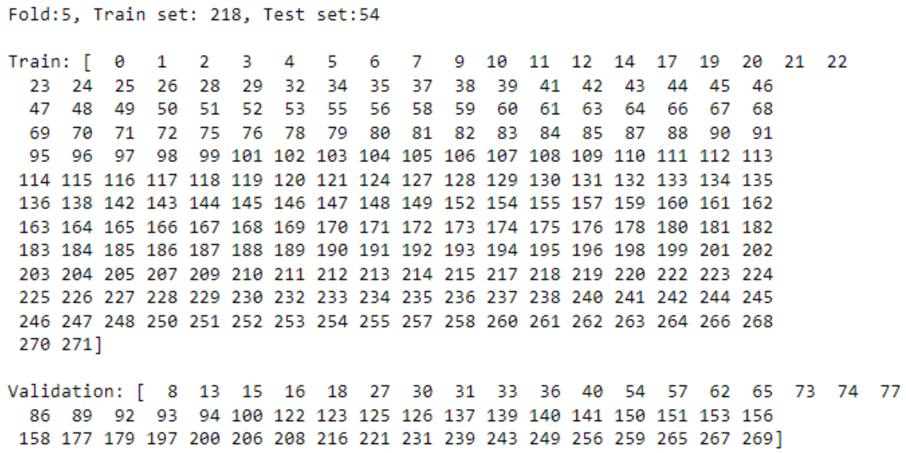
In this exercise the number of splits k = 5 , with Shuffle = True

There are the following folds:

Fold:1, Train set: 217, Test set:55
Fold:2, Train set: 217, Test set:55
Fold:3, Train set: 218, Test set:54
Fold:4, Train set: 218, Test set:54
Fold:5, Train set: 218, Test set:54



Example of the 5th fold [Indexes], Note that it has a random state = 10



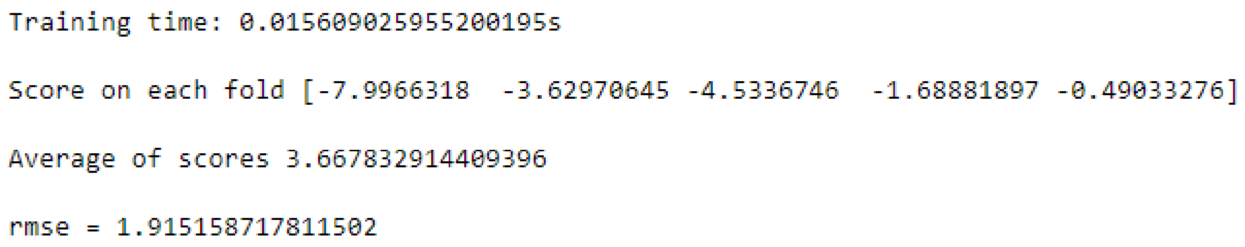
```
Fold:5, Train set: 218, Test set:54

Train: [  0   1   2   3   4   5   6   7   9  10  11  12  14  17  19  20  21  22
  23  24  25  26  28  29  32  34  35  37  38  39  41  42  43  44  45  46
  47  48  49  50  51  52  53  55  56  58  59  60  61  63  64  66  67  68
  69  70  71  72  75  76  78  79  80  81  82  83  84  85  87  88  90  91
  95  96  97  98  99 101 102 103 104 105 106 107 108 109 110 111 112 113
 114 115 116 117 118 119 120 121 124 127 128 129 130 131 132 133 134 135
 136 138 142 143 144 145 146 147 148 149 152 154 155 157 159 160 161 162
 163 164 165 166 167 168 169 170 171 172 173 174 175 176 178 180 181 182
 183 184 185 186 187 188 189 190 191 192 193 194 195 196 198 199 201 202
 203 204 205 207 209 210 211 212 213 214 215 217 218 219 220 222 223 224
 225 226 227 228 229 230 232 233 234 235 236 237 238 240 241 242 244 245
 246 247 248 250 251 252 253 254 255 257 258 260 261 262 263 264 266 268
 270 271]

Validation: [  8  13  15  16  18  27  30  31  33  36  40  54  57  62  65  73  74  77
  86  89  92  93  94 100 122 123 125 126 137 139 140 141 150 151 153 156
 158 177 179 197 200 206 208 216 221 231 239 243 249 256 259 265 267 269]
```

Linear Regression Not Normalized on Cross-Validation Set

On this set, the scoring is measured with the MSE, and then take the average of all five scores, then proceed to take the square root to obtain the RMSE.

“3.3.1.1. Common cases: predefined values For the most common use cases, you can designate a scorer object with the scoring parameter; the table below shows all possible values. All scorer objects follow the convention that higher return values are better than lower return values. Thus metrics that measure the distance between the model and the data, like metrics.mean_squared_error, are available as neg_mean_squared_error which returns the negated value of the metric.” (Buitinick et al., 2011)

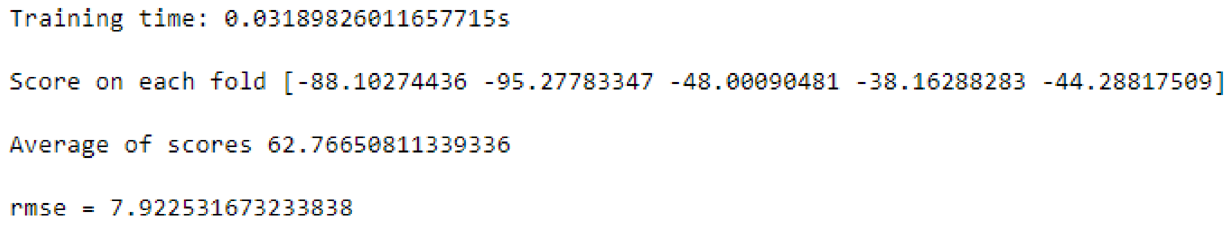
Seems that the cross-validation is not helping in this case, however, we will have to wait until the data has been normalized to see how it behaves.



```
Training time: 0.015609025955200195s

Score on each fold [-7.9966318  -3.62970645 -4.5336746  -1.68881897 -0.49033276]

Average of scores 3.667832914409396

rmse = 1.915158717811502
```

Decision Tree Regressor Not Normalized on Cross-Validation Set

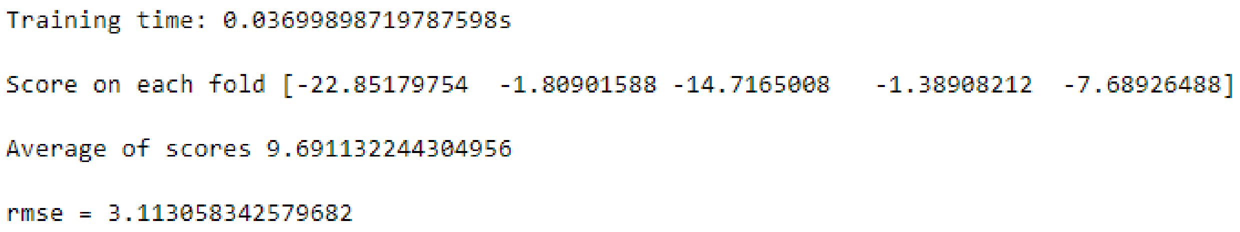
This regressor is still behaving very quite poor, even more than the Linear Regression CV,



```
Training time: 0.03189826011657715s

Score on each fold [-88.10274436 -95.27783347 -48.00090481 -38.16288283 -44.28817509]

Average of scores 62.76650811339336

rmse = 7.922531673233838
```

LASSO Regressor (Least Absolute Shrinkage and SelectionOperator) Not Normalized on Cross-Validation Set

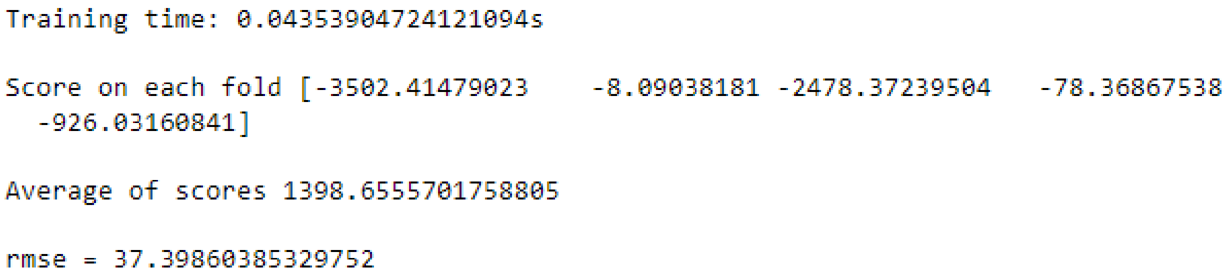
This regressor is still bad, better than the Decision Tree CV but worst than Linear Regression CV



```
Training time: 0.03699898719787598s

Score on each fold [-22.85179754  -1.80901588 -14.7165008   -1.38908212  -7.68926488]

Average of scores 9.691132244304956

rmse = 3.113058342579682
```

SVM Support Vector Machine Regressor Not Normalized on Cross-Validation Set

This regressor has the worst of all the RMSE’s, but as we know, SVM is very sensible tor Normalization



```
Training time: 0.04353904724121094s

Score on each fold [-3502.41479023    -8.09038181 -2478.37239504   -78.36867538
  -926.03160841]

Average of scores 1398.6555701758805

rmse = 37.39860385329752
```

Normalized on Cross-Validation Set

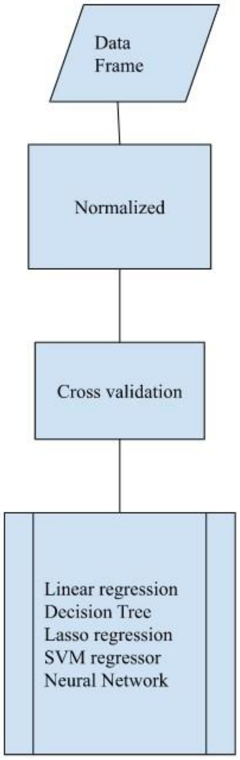
The following approach will be followed in this section.

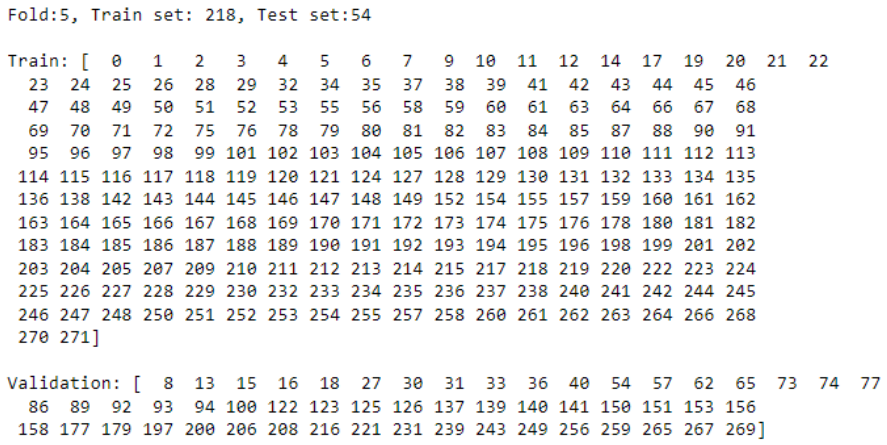
In this exercise the number of splits k = 5, with Shuffle = True

There are the following folds:

Fold:1, Train set: 217, Test set:55
Fold:2, Train set: 217, Test set:55
Fold:3, Train set: 218, Test set:54
Fold:4, Train set: 218, Test set:54
Fold:5, Train set: 218, Test set:54

Example of the 5th fold [Indexes]
Note that it has a random state = 10

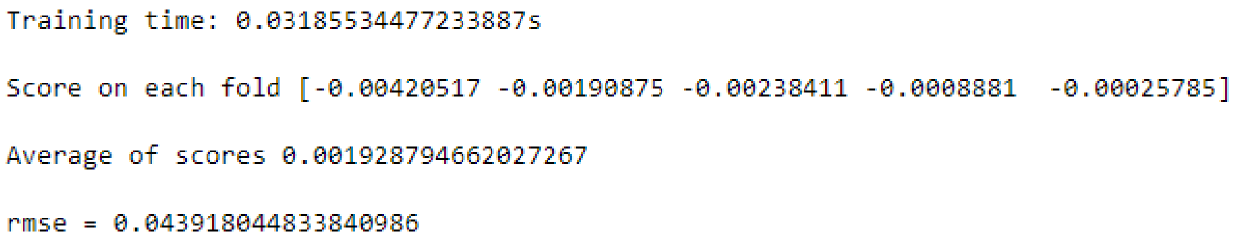




```
Fold:5, Train set: 218, Test set:54

Train: [  0   1   2   3   4   5   6   7   9  10  11  12  14  17  19  20  21  22
  23  24  25  26  28  29  32  34  35  37  38  39  41  42  43  44  45  46
  47  48  49  50  51  52  53  55  56  58  59  60  61  63  64  66  67  68
  69  70  71  72  75  76  78  79  80  81  82  83  84  85  87  88  90  91
  95  96  97  98  99 101 102 103 104 105 106 107 108 109 110 111 112 113
 114 115 116 117 118 119 120 121 124 127 128 129 130 131 132 133 134 135
 136 138 142 143 144 145 146 147 148 149 152 154 155 157 159 160 161 162
 163 164 165 166 167 168 169 170 171 172 173 174 175 176 178 180 181 182
 183 184 185 186 187 188 189 190 191 192 193 194 195 196 198 199 201 202
 203 204 205 207 209 210 211 212 213 214 215 217 218 219 220 222 223 224
 225 226 227 228 229 230 232 233 234 235 236 237 238 240 241 242 244 245
 246 247 248 250 251 252 253 254 255 257 258 260 261 262 263 264 266 268
 270 271]

Validation: [  8  13  15  16  18  27  30  31  33  36  40  54  57  62  65  73  74  77
  86  89  92  93  94 100 122 123 125 126 137 139 140 141 150 151 153 156
 158 177 179 197 200 206 208 216 221 231 239 243 249 256 259 265 267 269]
```

Linear Regression Normalized on Cross-Validation Set

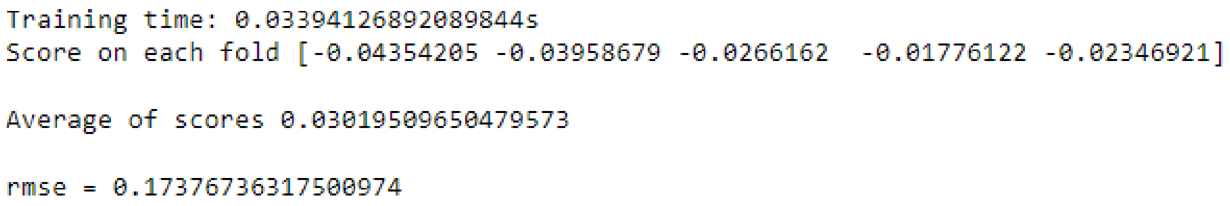
Here we have a much better RMSE, we are tending towards zero



```
Training time: 0.03185534477233887s

Score on each fold [-0.00420517 -0.00190875 -0.00238411 -0.0008881  -0.00025785]

Average of scores 0.001928794662027267

rmse = 0.043918044833840986
```

Decision Tree Regressor Normalized on Cross-Validation Set

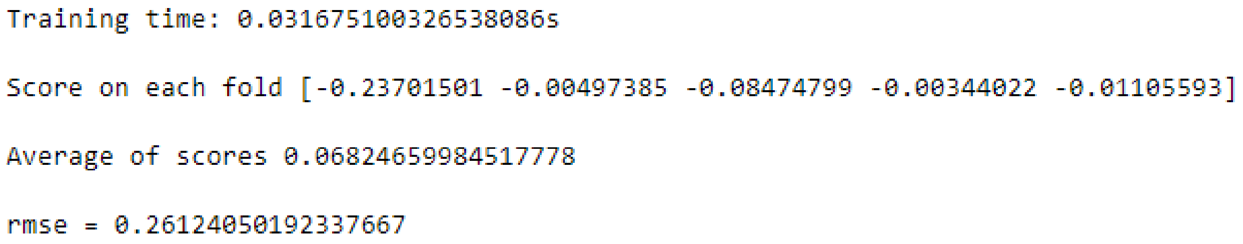
This algorithm has improved a lot when being Normalized, since the previous decision Tree CV had a RMSE of about 7



```
Training time: 0.03394126892089844s
Score on each fold [-0.04354205 -0.03958679 -0.0266162  -0.01776122 -0.02346921]

Average of scores 0.03019509650479573

rmse = 0.17376736317500974
```

LASSO Regressor (Least Absolute Shrinkage and SelectionOperator) Normalized on Cross-Validation Set

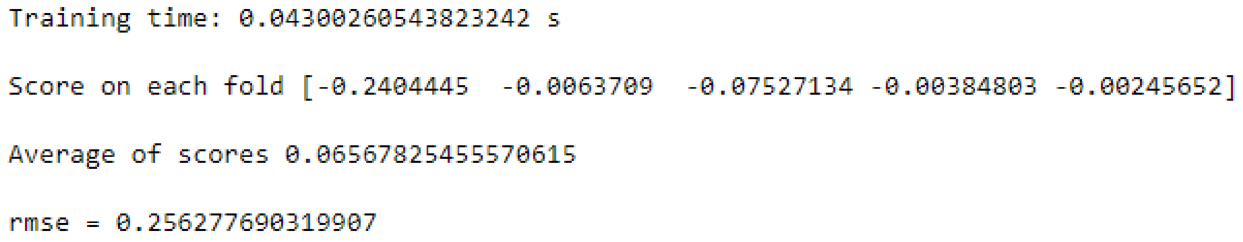
The algorithm is performing really well since the RMSE is tending towards zero



```
Training time: 0.031675100326538086s

Score on each fold [-0.23701501 -0.00497385 -0.08474799 -0.00344022 -0.01105593]

Average of scores 0.06824659984517778

rmse = 0.26124050192337667
```

SVM Support Vector Machine Regressor Normalized on Cross-Validation Set

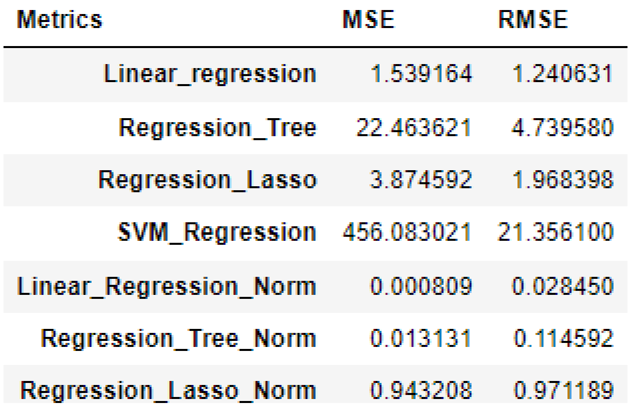
This regressor had a RMSE of 37.39 and has improved all the way to 0.25, this confirms that is always best to normalize the data in order to obtain much better results



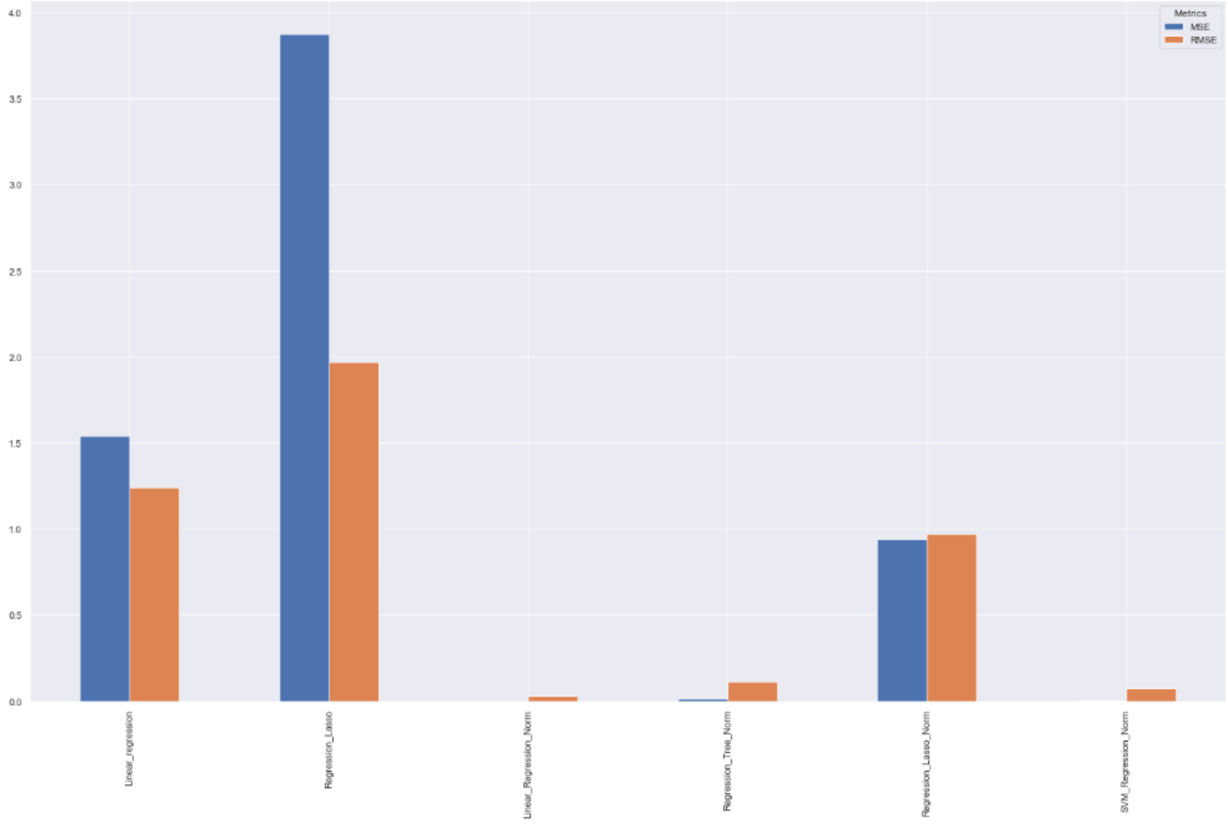
```
Training time: 0.04300260543823242 s

Score on each fold [-0.2404445  -0.0063709  -0.07527134 -0.00384803 -0.00245652]

Average of scores 0.06567825455570615

rmse = 0.256277690319907
```

Effectiveness in Train set and Test set

Once the code has been executed, the results can be seen on the following table.
This table contains the values of the data frame **Normalized** and **Not Normalization**, and the MSE (Mean Squared Error), as well as the RMSE (Root Mean Squared Error), have been measured.



Metrics	MSE	RMSE
Linear_regression	1.539164	1.240631
Regression_Tree	22.463621	4.739580
Regression_Lasso	3.874592	1.968398
SVM_Regression	456.083021	21.356100
Linear_Regression_Norm	0.000809	0.028450
Regression_Tree_Norm	0.013131	0.114592
Regression_Lasso_Norm	0.943208	0.971189

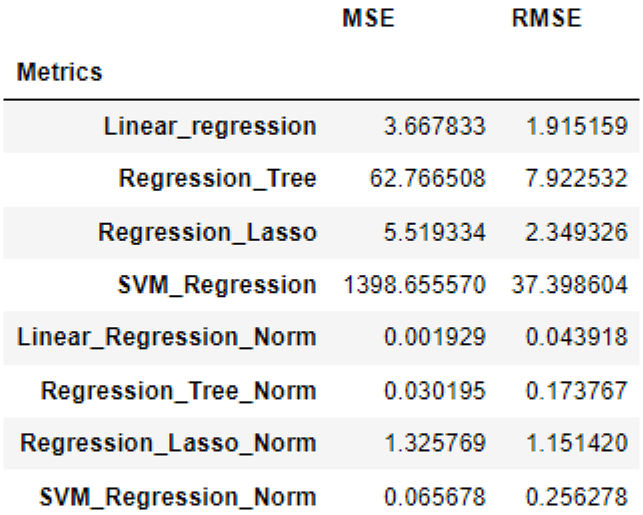


Can be seen that the best algorithm when having a training set and testing set is Linear regression Normalized followed by SVM Regression Normalized.

Note that SVM Regression and Regression Tree had to be dropped as they both have very high numbers, and once they have been normalized they perform really well, which explains why it is good to normalize when doing linear regression problems.

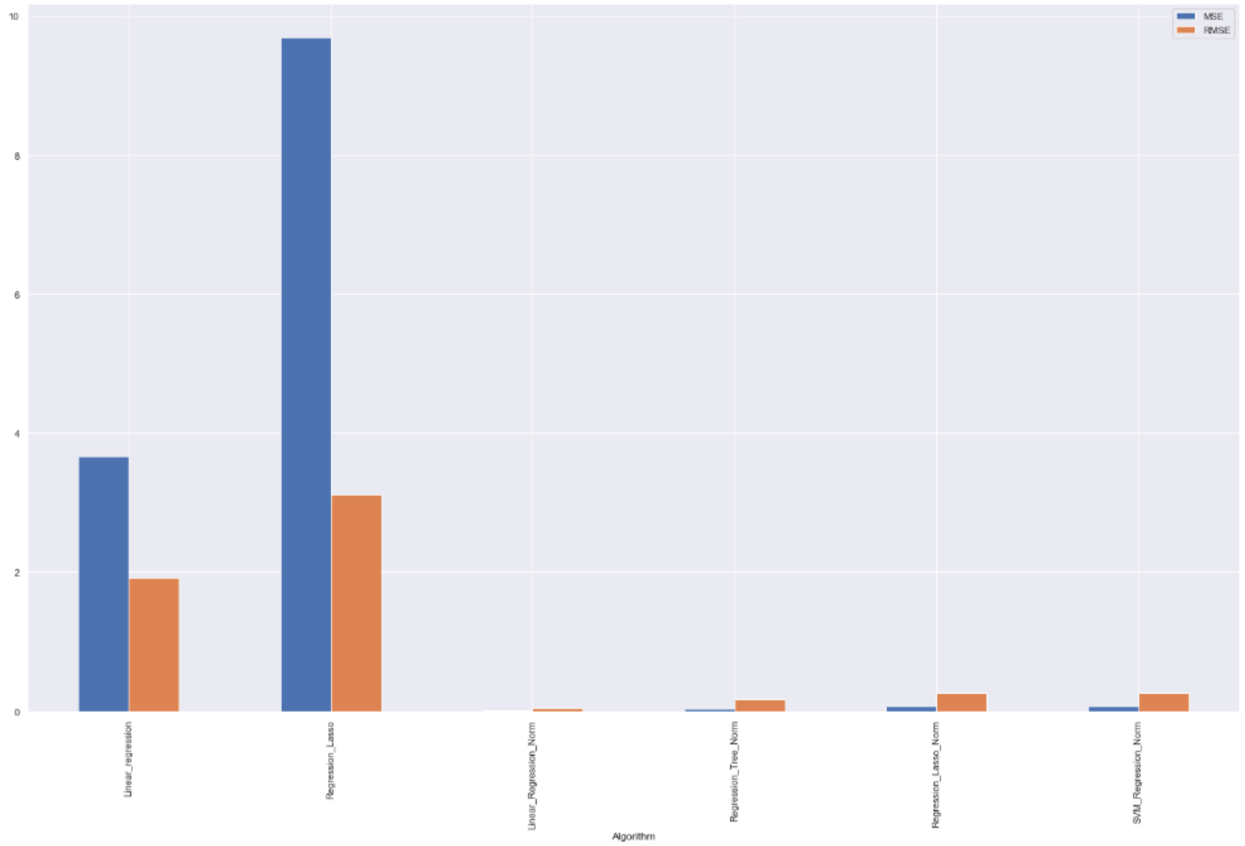
Effectiveness in Cross Validation

When applying the same algorithms to the data frame using Cross Validation, and **Normalized** and **Not Normalized** we obtain the following table

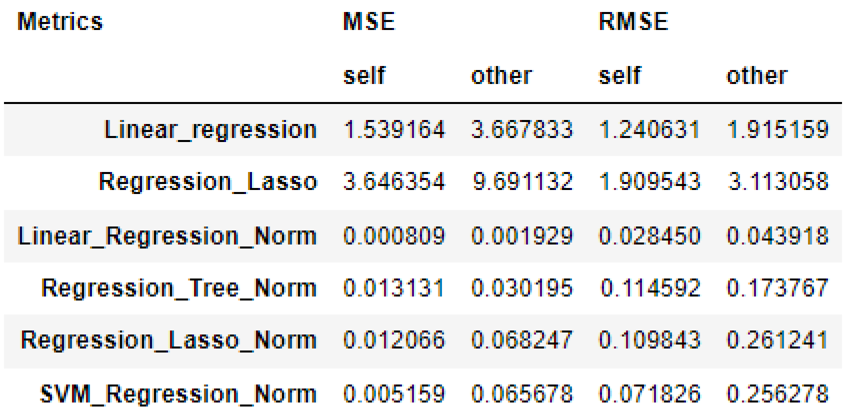


	MSE	RMSE
Metrics		
Linear_regression	3.667833	1.915159
Regression_Tree	62.766508	7.922532
Regression_Lasso	5.519334	2.349326
SVM_Regression	1398.655570	37.398604
Linear_Regression_Norm	0.001929	0.043918
Regression_Tree_Norm	0.030195	0.173767
Regression_Lasso_Norm	1.325769	1.151420
SVM_Regression_Norm	0.065678	0.256278

Can be seen that the most effective algorithm when doing Cross Validation, is the algorithm of Linear Regression Normalized, followed by Regression tree Normalized, note that SVM Regression and Regression Tree are very bad when not normalized, but Regression tree becomes second best once it has been normalized.



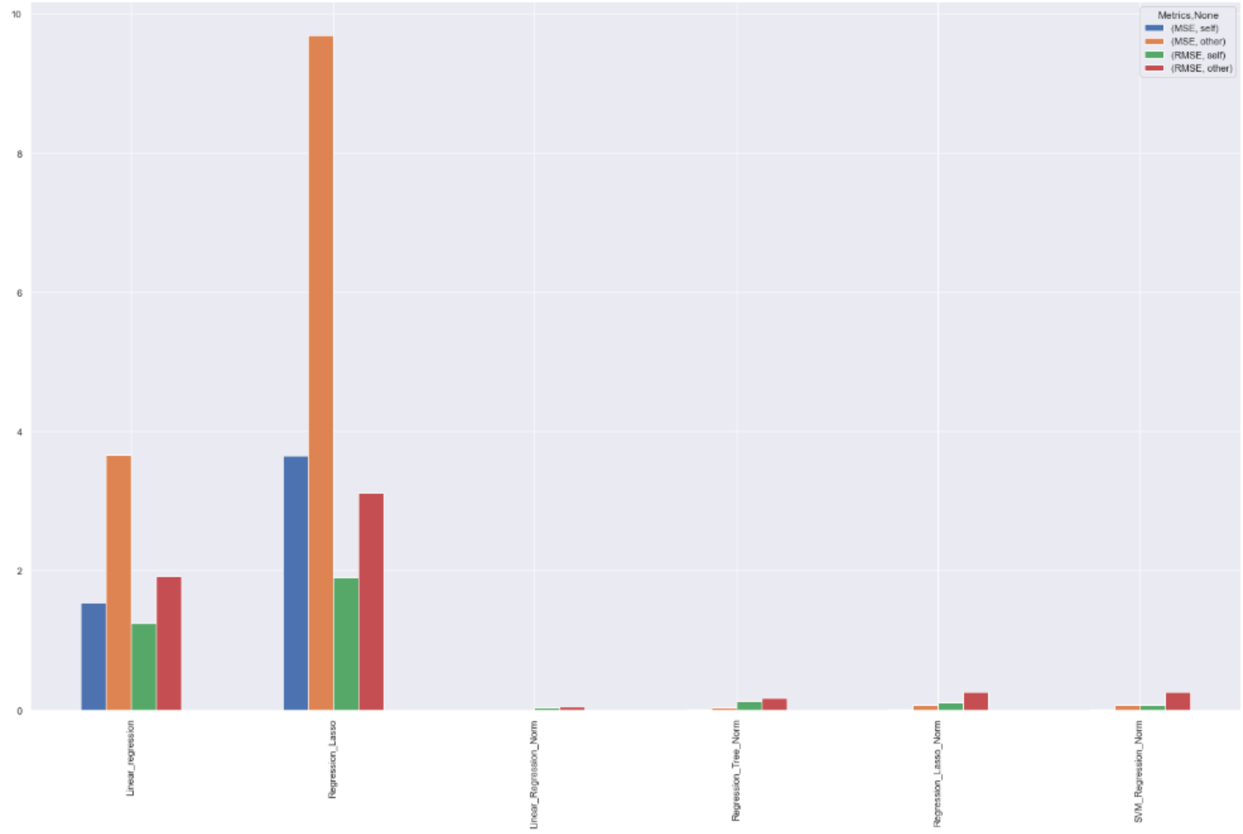
When comparing both results side by side, the following table is obtained



Metrics	MSE		RMSE	
	self	other	self	other
Linear_regression	1.539164	3.667833	1.240631	1.915159
Regression_Lasso	3.646354	9.691132	1.909543	3.113058
Linear_Regression_Norm	0.000809	0.001929	0.028450	0.043918
Regression_Tree_Norm	0.013131	0.030195	0.114592	0.173767
Regression_Lasso_Norm	0.012066	0.068247	0.109843	0.261241
SVM_Regression_Norm	0.005159	0.065678	0.071826	0.256278

And can be seen from the graph, that the best algorithm to evaluate the model is the Linear Regression Normalized as it has the minimum values in both exercises.

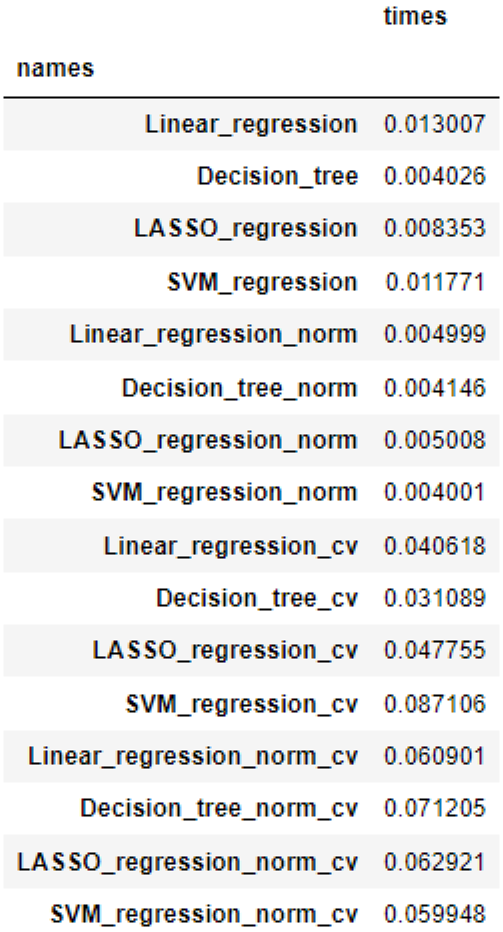
Please note that SVM_Regression and Regression_Tree have been dropped from the graph due to their high values as they will make disappear the other values on the graph.



Efficiency or Timing of the algorithms

In order to measure the time each model takes to be trained or tested, a data frame was created with the values of each algorithm when being timed.

The following table shows the times of each algorithm

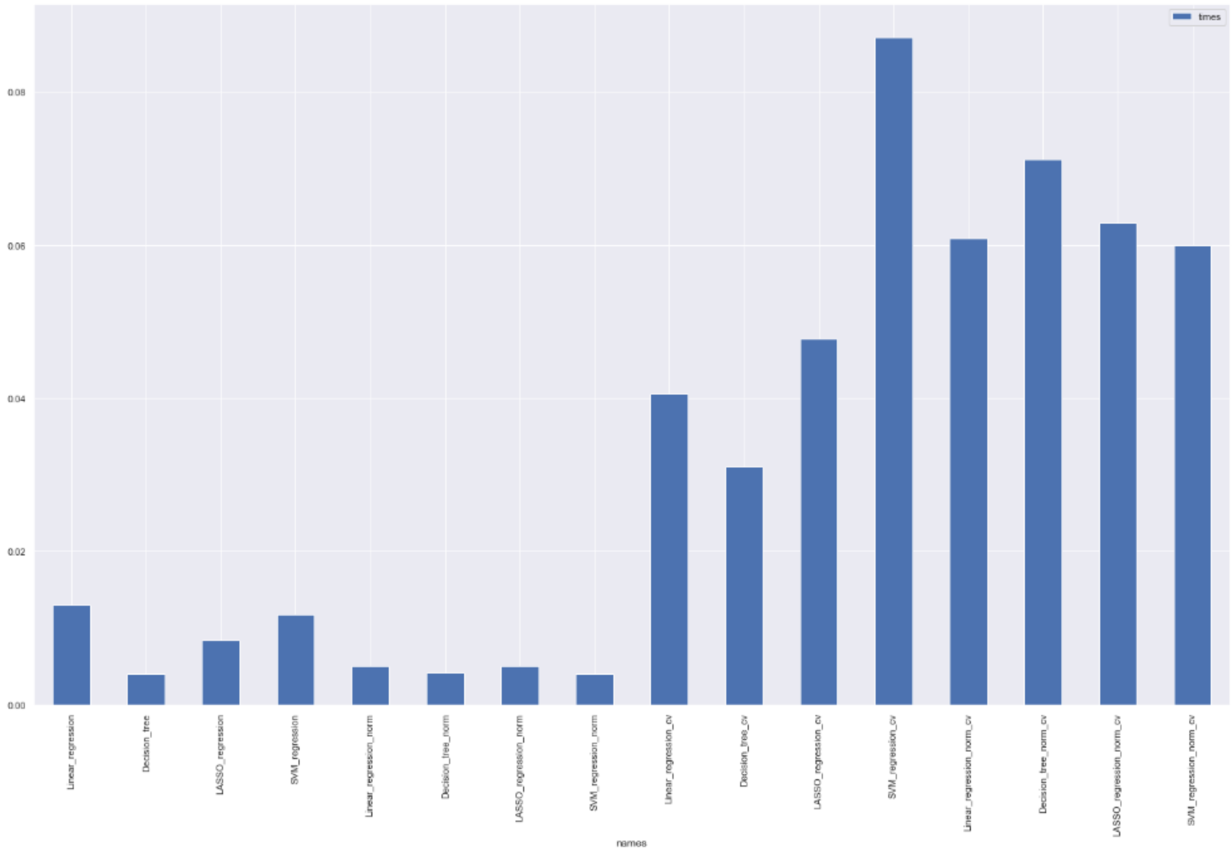


	times
names	
Linear_regression	0.013007
Decision_tree	0.004026
LASSO_regression	0.008353
SVM_regression	0.011771
Linear_regression_norm	0.004999
Decision_tree_norm	0.004146
LASSO_regression_norm	0.005008
SVM_regression_norm	0.004001
Linear_regression_cv	0.040618
Decision_tree_cv	0.031089
LASSO_regression_cv	0.047755
SVM_regression_cv	0.087106
Linear_regression_norm_cv	0.060901
Decision_tree_norm_cv	0.071205
LASSO_regression_norm_cv	0.062921
SVM_regression_norm_cv	0.059948

The **slowest** algorithm is SVM Regression CV when doing Cross-Validation and the same algorithm when being normalized and using the same Cross-Validation technique has a time that is much less

The **fastes**t algorithm is SVM regression Normalized when doing the exercise with Train and test validation sets, and the same algorithm when doing Cross validation became the slowest

The graph of all the times can be seen next

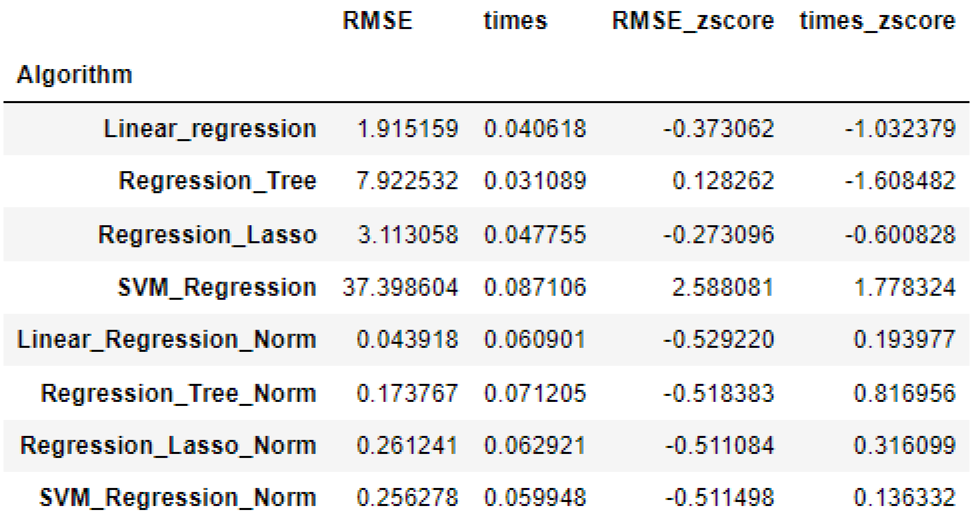


Note that there is a difference using cross-validation vs Training and testing set, where using Cross-validation increases the timing of the algorithms by almost more than double

The best Algorithm

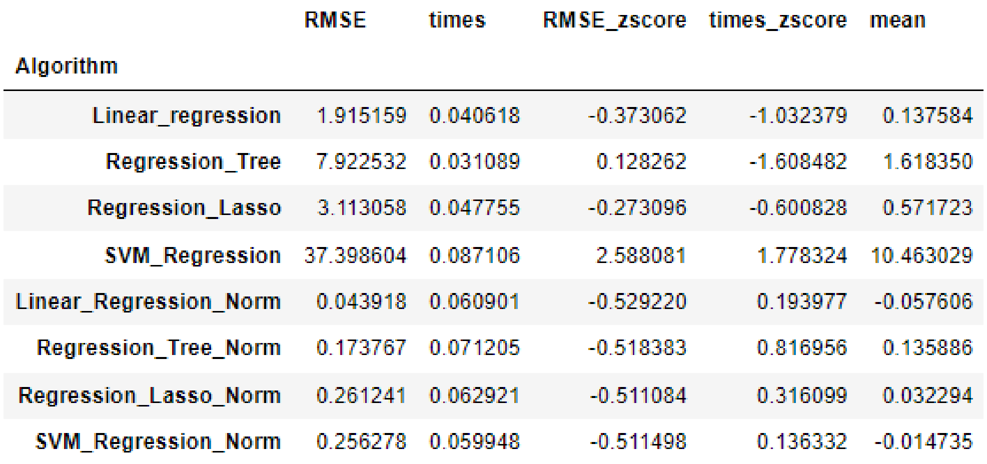
Well, we have measured the time and the effectiveness, and now is the time to obtain the best algorithm based on these results, however, as a machine learning algorithm, it can change or obtain different results every time it is run.

For the exercise of Cross-validation, I’ll use RMSE as the metric and the time in seconds. I create a data frame that contains RMSE and the times and proceed to escalate it using Z-Score, the data frame can be seen next:



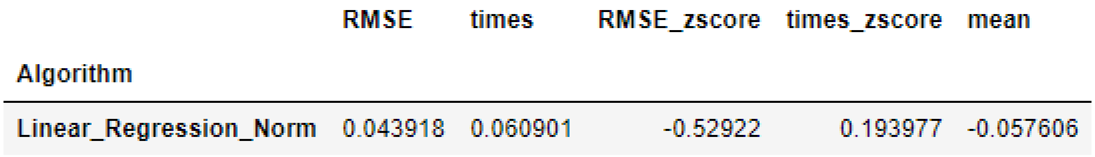
	RMSE	times	RMSE_zscore	times_zscore
Algorithm				
Linear_regression	1.915159	0.040618	-0.373062	-1.032379
Regression_Tree	7.922532	0.031089	0.128262	-1.608482
Regression_Lasso	3.113058	0.047755	-0.273096	-0.600828
SVM_Regression	37.398604	0.087106	2.588081	1.778324
Linear_Regression_Norm	0.043918	0.060901	-0.529220	0.193977
Regression_Tree_Norm	0.173767	0.071205	-0.518383	0.816956
Regression_Lasso_Norm	0.261241	0.062921	-0.511084	0.316099
SVM_Regression_Norm	0.256278	0.059948	-0.511498	0.136332

Then, once it has been Z-scored, the program will calculate the mean of each row (RMSE and Time), and then it will take the minimum value of all the averages, and this is the algorithm that best performs using RMSE and Time.



	RMSE	times	RMSE_zscore	times_zscore	mean
Algorithm					
Linear_regression	1.915159	0.040618	-0.373062	-1.032379	0.137584
Regression_Tree	7.922532	0.031089	0.128262	-1.608482	1.618350
Regression_Lasso	3.113058	0.047755	-0.273096	-0.600828	0.571723
SVM_Regression	37.398604	0.087106	2.588081	1.778324	10.463029
Linear_Regression_Norm	0.043918	0.060901	-0.529220	0.193977	-0.057606
Regression_Tree_Norm	0.173767	0.071205	-0.518383	0.816956	0.135886
Regression_Lasso_Norm	0.261241	0.062921	-0.511084	0.316099	0.032294
SVM_Regression_Norm	0.256278	0.059948	-0.511498	0.136332	-0.014735

Through almost all the exercises, the algorithm Linear Regression and Linear Regression Normalized look like the best answer, in the end, is Linear Regression Normalized the one that has the highest score to predict the algorithm base on Time and RMSE.



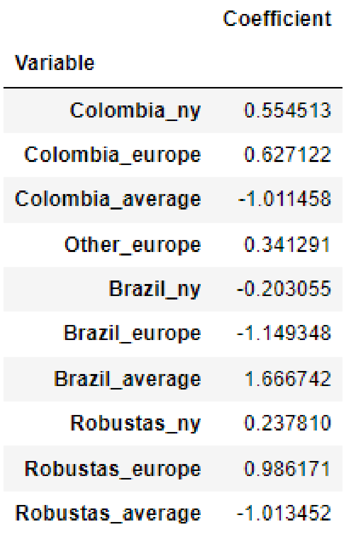
	RMSE	times	RMSE_zscore	times_zscore	mean
Algorithm					
Linear_Regression_Norm	0.043918	0.060901	-0.52922	0.193977	-0.057606

Neural Network

Neural network is an exercise that I have done as a personal challenge and to gain knowledge on this interesting topic.

In order to do a neural Network, experts recommend normalizing the data and since it has been normalized then I will use the previously normalized data.

The first step is to obtain the coefficients of the model in this case:



	Coefficient
Variable	
Colombia_ny	0.554513
Colombia_europe	0.627122
Colombia_average	-1.011458
Other_europe	0.341291
Brazil_ny	-0.203055
Brazil_europe	-1.149348
Brazil_average	1.666742
Robustas_ny	0.237810
Robustas_europe	0.986171
Robustas_average	-1.013452

Then the model is saved as a numpy with the 2 sets of data inputs = X and OIC_price = y

Then the number of input layers is 12 as there are 12 columns in the data set in order to obtain the variable that we need, in this case is the column OIC_price, and this is the output layer, or basically what we want to predict.

A dense layer is needed (from all 12 neurons to one neuron, or basically from all 12 columns to the predicted column OIC_price)

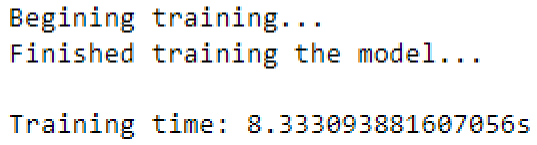
Then I have to build a model for the layers, in this case, the Sequential (a sequence of layers stacked one after another) that is for a neural network that is not advanced.

I have to prepare the model to be trained (this is called a compiler), and I will use an optimizer called Adam, which allows the network to know how to use the bias and weights in an efficient way, so it learns, instead of unlearning (gets better step by step).

The learning rate is 0.1, which indicates how to adjust the weights and bias (increasing in steps of 0.1)

To train the model, I choose epochs = 400, this is how many times I want it to loop (1 loop means checking all 272 entries), usually the more epochs the better training but only until a certain point.

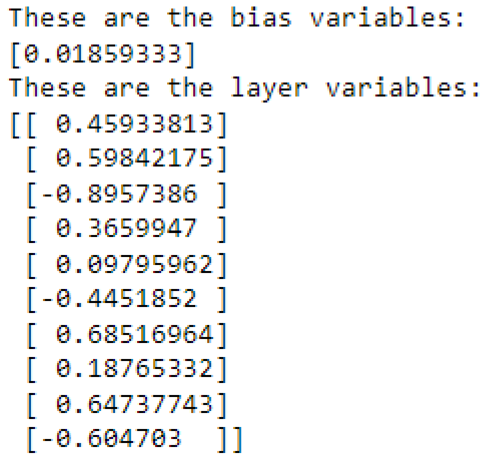
The verbose = false, so it doesn't print anything



```
Begining training...
Finished training the model...

Training time: 8.333093881607056s
```

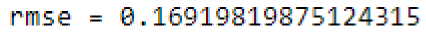
The algorithm trained the model in 8.3 seconds

then I want to see the Weights and the Bias



```
These are the bias variables:
[0.01859333]
These are the layer variables:
[[ 0.45933813]
 [ 0.59842175]
 [-0.8957386 ]
 [ 0.3659947 ]
 [ 0.09795962]
 [-0.4451852 ]
 [ 0.68516964]
 [ 0.18765332]
 [ 0.64737743]
 [-0.604703  ]]
```

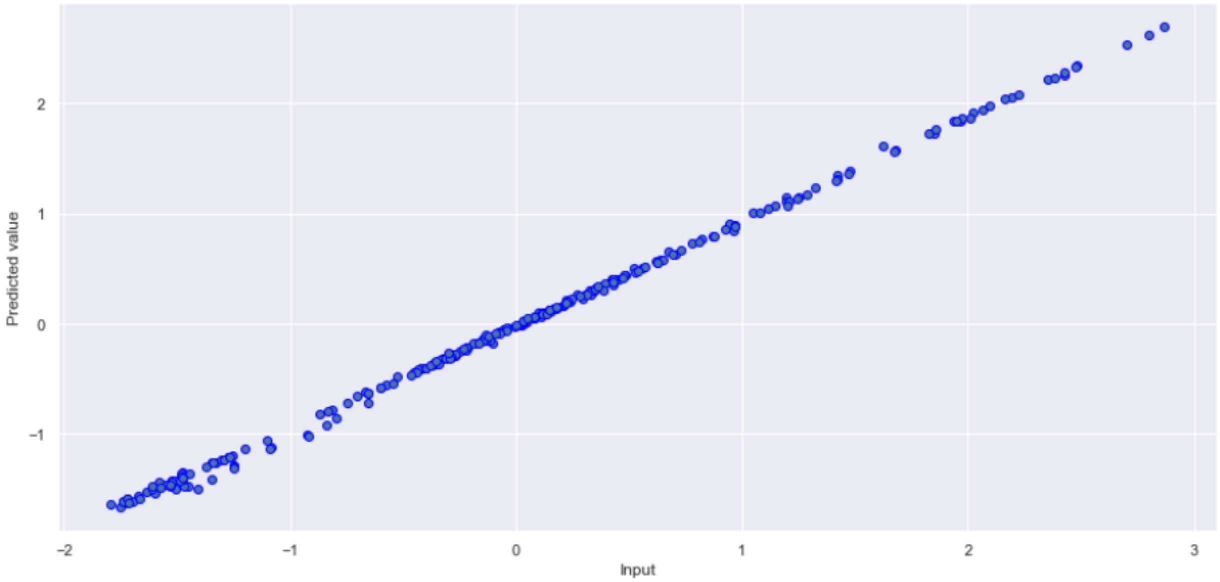
And I want to measure the Root Mean Squared Error RMSE



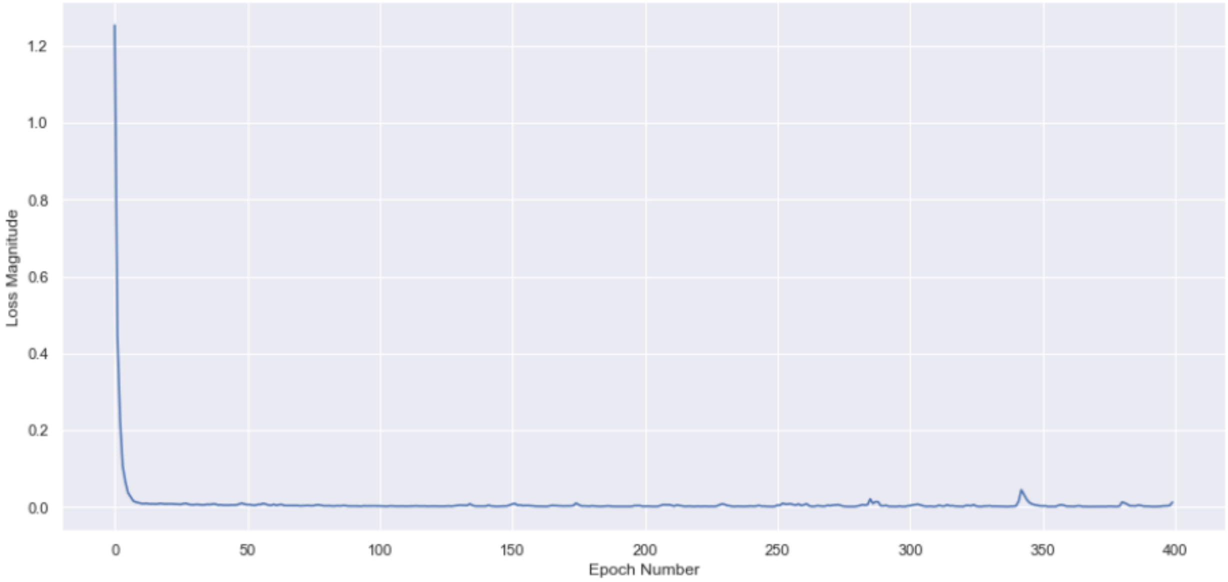
```
rmse = 0.16919819875124315
```

Which is a very good RMSE, but not as good as the Linear Regression or the other Algorithms.

The plot of the graph of Predicted Values vs Input can be seen next



To measure the loss or basically how bad are the results with each loop that was done



From here I can say that the system could have been trained with much fewer loops or epochs

References

Buitinick, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., & Varoquaux, G. (2011). *3.3. Metrics and scoring: quantifying the quality of predictions*. Scikit-learn user Guide. Retrieved November 19, 2022, from https://scikit-learn.org/stable/modules/model_evaluation.html#scoring-parameter%22

Deina, C., do Amaral Prates, M. H., Rodrigues Alves, C. H., Ribeiro Martins, M. S., Trojan, F., Stevan Jr, S. L., & Valares Siqueira, H. (2022, September 27). A methodology for coffee price forecasting based on extreme learning machines. *Information Processing in Agriculture*, *9*(3), 10. https://www.sciencedirect.com/science/article/pii/S2214317321000597

Diario el Mundo. (2021, February 15). ¿Cuál es el panorama mundial de la producción y del precio del café en 2021? *Diario El Mundo*. https://diario.elmundo.sv/Econom%C3%ADa/cual-es-el-panorama-mundial-de-la-produccion-y-del-precio-del-cafe-en-2021

E. Sreehari and S. Srivastava, "Prediction of Climate Variable using Multiple Linear Regression," 2018 4th International Conference on Computing Communication and Automation (ICCCA), 2018, pp. 1-4, doi: 10.1109/CCAA.2018.8777452.

Federación Nacional de Cafeteros de Colombia. (2021). La cosecha asistida de café y su impacto en la economía de la recolección en finca. Ensayos sobre Economía Cafetera, 34(1), 35-50

Federación Nacional de Cafeteros de Colombia. (2022, septiembre 16). Precios, área y producción del café. Bogota; https://federaciondecafeteros.org/app/uploads/2020/01/Precios-%C3%A1rea-y-producci%C3%B3n-de-caf%C3%A9-3.xlsx.

Gil Serna, J. G. (2012). *ESTIMACIÓN DE UN PRONÓSTICO DE EXPORTACIONES DE CAFÉ SUAVE COLOMBIANO: REDES NEURONALES ARTIFICIALES Y ARDL*.

H. L. Siew and M. J. Nordin, "Regression techniques for the prediction of stock price trend," 2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE), 2012, pp. 1-5, doi: 10.1109/ICSSBE.2012.6396535.

ICO.ORG. (2014, March 11). MONTHLY COFFEE MARKET REPORT - February 2014. *International Coffee Organization*. http://www.ico.org/documents/cy2013-14/cmr-0214-e.pdf

Kavitha S, Varuna S and Ramya R, "A comparative analysis on linear regression and support vector regression," 2016 Online International Conference on Green Engineering and Technologies (IC-GET), 2016, pp. 1-5, doi: 10.1109/GET.2016.7916627.

K. M. Hindrayani, T. M. Fahrudin, R. Prismahardi Aji and E. M. Safitri, "Indonesian Stock Price Prediction including Covid19 Era Using Decision Tree Regression," 2020 3rd

International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), 2020, pp. 344-347, doi: 10.1109/ISRITI51436.2020.9315484.

Li, R., Laroche, M., Richard, M. O., & Cui, X. (2022, January 24). More than a mere cup of coffee: When perceived luxuriousness triggers Chinese customers' perceptions of quality and self-congruity. *Journal of Retailing and Consumer Services*, *64*(102759). https://www.sciencedirect.com/science/article/abs/pii/S0969698921003258

Sanz Uribe, J. r., Duque Orrego, H., Gaitan Bustamante, A. Federación Nacional de Cafeteros de Colombia. (2021, June 25). La cosecha asistida de café y su impacto en la economía de la recolección en finca. *Ensayos sobre Economía Cafetera*, 35-50.

Statistics Committee. (2021, March 29). sc-106e-rules-indicator-prices.pdf. London; International Coffee Organization.

Suarez Pena, J. A. (2019, December). *MODELO DE APRENDIZAJE AUTOMÁTICO PARA LA PREDICCIÓN DE LA CALIDAD DEL CAFÉ.* [Proyecto de Grado para obtener el título de Magister en Ingeniería Industrial]. Repositorio Institucional Universidad Distrital - RIUD. https://repository.udistrital.edu.co/bitstream/handle/11349/23560/SuarezPe%C3%B1aJavierAndres2019.pdf?sequence=1&isAllowed=y

Wile, R. (2014, February 24). *Why Coffee Prices Are Surging*. Business Insider. Retrieved October 23, 2022, from https://www.businessinsider.com/why-coffee-prices-are-surging-2014-2?r=US&IR=T