PROYECTO FINAL
MACHINE LEARNING I

Housing in Austin

5 DE ENERO DE 2024

SANTIAGO ARENAS
Santiago Arenas

Global Goals

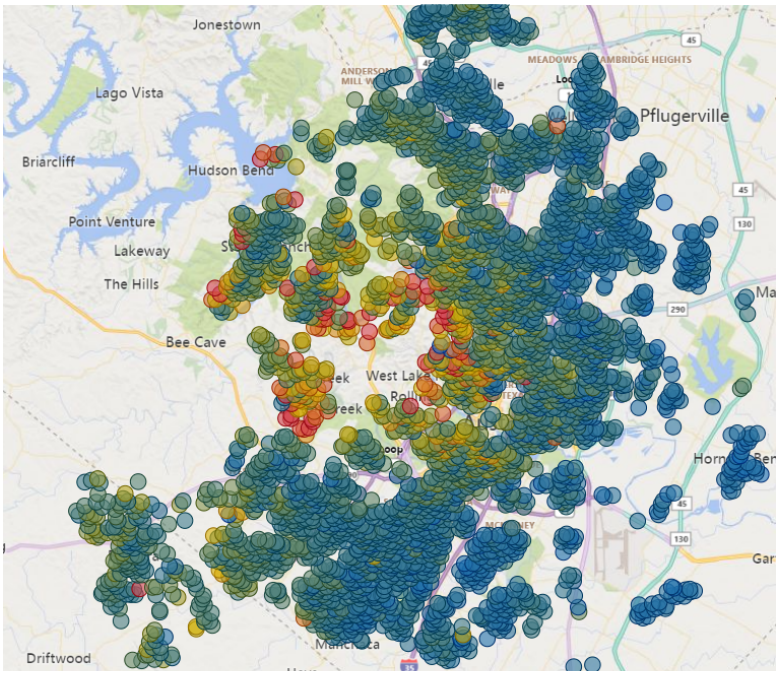
The main objective of this project is to obtain a regressor that allows us to have a reference price when buying or selling a house in Austin, Texas. We also want to obtain which are the characteristics that are most valued (in the appraisal) in this city for decision-making in a possible reform focused on increasing the value of a house. In addition, we will try to work in the field of classification to find conclusions that may be of interest for the analysis of this real estate market.
This regressor could avoid scams, or in the opposite case, it could find market opportunities. We do not expect to find a model that predicts the price of a home with great accuracy, but we do expect to find a good and robust enough to be able to assess and make a first pre-selection of the offers.

Methodology followed

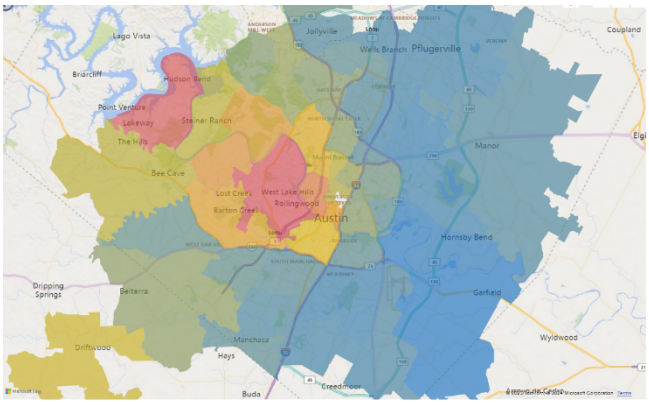
We will use the knowledge and algorithms studied throughout the course to maximize the learning of the cases at our disposal. To compare the different regressors and determine which is the best, we establish the RMSE as a comparison parameter.
We will follow the development of the different algorithms as well as comment on the changes and decisions made at all times. In turn, throughout this development we will highlight conclusions and relevant information resulting from the algorithms.
We will start from the initial data that we will present below, but at any time we will assess whether a modification is necessary, either by artificially adding or removing values or certain specific cases.

Presentation of data

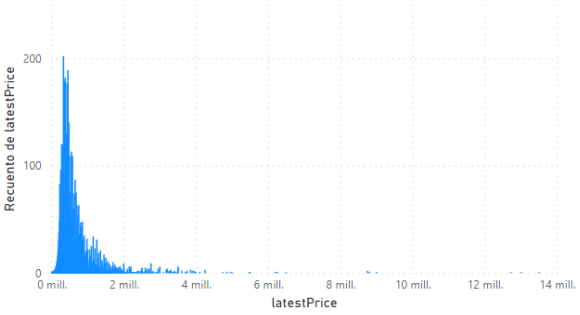
We start from a Kaggle dataset whose size is more than 15000 cases and 47 columns. This file collects data on home sales in Austin, Texas from 2018 to 2021. To understand the distribution of the data we will use the Power BI tool:

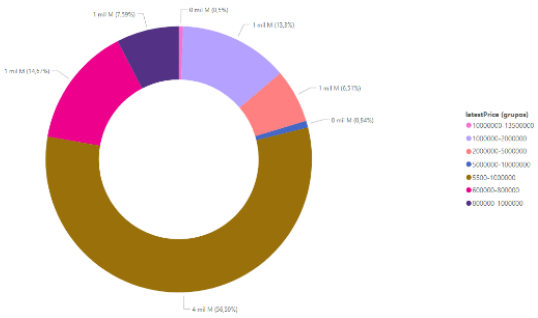


On the one hand, looking at the heat and bubble maps, you can see that some areas are more expensive than others. As in all cities, it can be seen that the center is one of the most expensive areas, but the part near the Colorado River is also added where there is also an upturn in house prices. Therefore, estimating from the outset that the area of the house influences the price can be correct. Even so, we see how there are some specific houses with a high price in the surroundings that surely do not facilitate the analysis with our algorithms and we expect the appearance of extreme values like these that we will have to deal with.

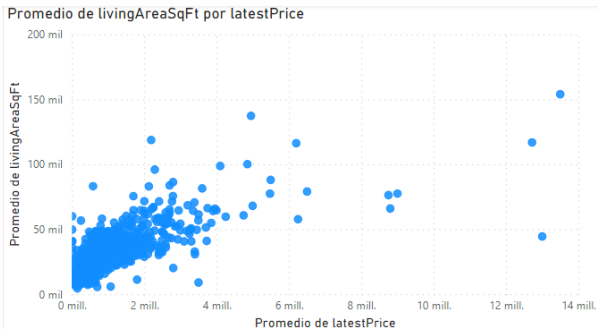


On the other hand, looking at the following graphs we can see the distributions of the data. The data is mostly at 5500 and 2 million dollars. Despite this, there are very large and very distant values (outliers/extreme values) that we will see how they affect our models.





In addition, we come up with several relationships that we hope to see later in our models. Like the price being proportional to the habitable meters as we see in the scatter plot. This relationship is usually fulfilled as a general rule, so it would be expected to see this variable as one of the most significant. There are many more variables and many more relationships that we forget to mention but that we will find throughout the development of the different algorithms.



Algorithms

Multiple Linear Regression (MLR)

Objectives

The main objective will be to obtain the best possible regressor from this algorithm, our variable to predict will be "LatestPrice".
With this first regressor we are looking for a first view of the data set, relationships between variables and, of course, to see how good this linear algorithm can be (with the possibility of adding interactions) in our particular case.
In addition, as we have seen in the brief previous analysis, there are certain price values that stand out and are very different from the others, so in this case we will consider making a model for a lower price range than the original. In this way, knowing the simplicity of this model, we can obtain a better model for a usual price range.

Expected results

As mentioned above, the multiple linear regression algorithm has difficulty with more complex relationships between variables. Therefore, in this first approximation we expect a simple model that is able to work well with the most common price ranges and provides us with a lot of information from our dataset.
As for the variables, we expect some to be much more significant than others. By common sense, we expect them to be important in our model.

Pre-processed

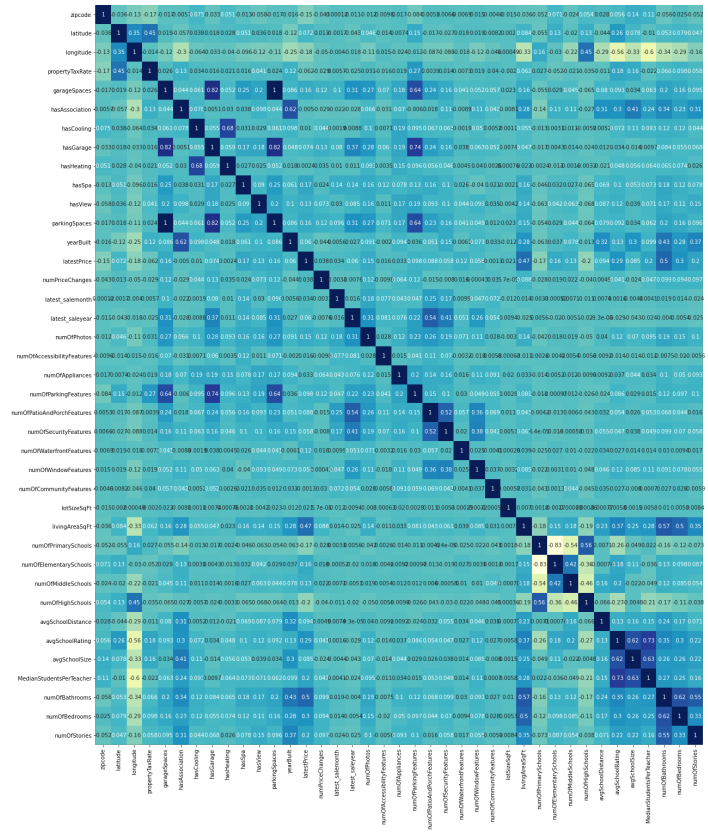
First we review the variables and their meaning to see if we will use all the columns from the beginning or if there are any that we can leave aside.

As a general rule for all regression algorithms, we will discard the columns "description" (textual description of the house), "homeImage" (link to the website), "zpid" (Zillow website id), "latestPriceSource" (source where it comes from), "latest_salesdate" (date of the last sale) and "streetAddress" (number and street of the house) that without doing any numerical analysis we can eliminate since they do not provide information ("homeImage", "latestPriceSource" and "zpid") or we cannot use it as it is written text and/or is explained by other variables ("latest_salesdate", "description" and "streetAddress").

In addition, for the price regressors we will take the values of the city of Austin only and of the different types of houses only those of single family ("hometype"). This is because, on the one hand, we are not interested in cases outside the city and, above all, the number of cases other than these are negligible.

We start with the data once we have eliminated the unwanted columns already mentioned. With these discards we still have more than 14,000 cases and 40 columns so we keep more than enough information.

Before we start modelling, we need to look at the different correlations between variables. With a correlation matrix we can estimate the following:



Here the relationships between all numerical and boolean variables are represented. There are too many to comment on them all, but we can mention some that stand out. For example, "parkingSpaces" is directly proportional to "garageSpaces" which tells us that the numeric values of these columns are identical so we can delete one of the two. We decided to remove "garageSpaces" simply because the data source told us that it was a subset of the other variable (but it wouldn't matter). Another correlation worth mentioning is the one between "MedianStudentsPerTeacher" and "avgSchoolSize" which gives a value greater than 0.8, which makes sense since the fact that there are more teachers on average helps to make that school better. The relationship between educational centers of different levels is also surprising, we see how in some cases it is inversely proportional, which can be attributed to their distribution throughout the city.

Finally, we must transform our Boolean variables into numerical variables, then we obtain the dummies. In addition, we divide our dataset into train and test, we choose 80% and 20% respectively.

Development

With all this we can already elaborate our first multiple linear regression model.

OLS Regression Results

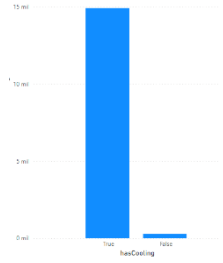
Dep. Variable:	latestPrice	R-squared:	0.528
Model:	OLS	Adj. R-squared:	0.527
Method:	Least Squares	F-statistic:	339.9
Date:	Tue, 02 Jan 2024	Prob (F-statistic):	0.00
Time:	21:26:55	Log-Likelihood:	-1.5861e+05
No. Observations:	11276	AIC:	3.173e+05
Df Residuals:	11238	BIC:	3.176e+05
Df Model:	37		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	3.742e+08	2e+07	18.730	0.000	3.35e+08	4.13e+08
zipcode	-4405.8477	214.390	-20.551	0.000	-4826.090	-3985.606
latitude	-3.416e+05	4.87e+04	-7.012	0.000	-4.37e+05	-2.46e+05
longitude	7.758e+05	6.69e+04	11.588	0.000	6.45e+05	9.07e+05
propertyTaxRate	-1.036e+06	7.54e+04	-13.751	0.000	-1.18e+06	-8.89e+05
parkingSpaces	1015.8588	4328.868	0.235	0.814	-7469.481	9501.198
yearBuilt	-844.2023	200.161	-4.218	0.000	-1236.553	-451.852
numPriceChanges	-1.234e+04	1224.001	-10.081	0.000	-1.47e+04	-9940.234
latest_salemonth	2256.5068	1006.112	2.243	0.025	284.352	4228.662
latest_saleyear	3.097e+04	4842.989	6.395	0.000	2.15e+04	4.05e+04
numOfPhotos	112.4690	156.063	0.721	0.471	-193.442	418.380
numOfAccessibilityFeatures	6268.9602	1.52e+04	0.413	0.679	-2.35e+04	3.6e+04
numOfAppliances	-53.7245	1675.286	-0.032	0.974	-3337.579	3230.130
numOfParkingFeatures	3.911e+04	6314.007	6.194	0.000	2.67e+04	5.15e+04
numOfPatioAndPorchFeatures	4352.3401	4061.835	1.072	0.284	-3609.568	1.23e+04
numOfSecurityFeatures	66.3201	4448.930	0.015	0.988	-8654.362	8787.002
numOfWaterfrontFeatures	5.972e+05	4.58e+04	13.047	0.000	5.07e+05	6.87e+05
numOfWindowFeatures	-1.702e+04	6740.100	-2.525	0.012	-3.02e+04	-3808.503
numOfCommunityFeatures	-7.338e+04	2.5e+04	-2.938	0.003	-1.22e+05	-2.44e+04
lotSizeSqFt	0.0003	0.000	1.418	0.156	-0.000	0.001
livingAreaSqFt	260.4239	5.371	48.486	0.000	249.896	270.952
numOfPrimarySchools	1.012e+05	2.82e+04	3.591	0.000	4.59e+04	1.56e+05
numOfElementarySchools	1.497e+05	2.41e+04	6.200	0.000	1.02e+05	1.97e+05
numOfMiddleSchools	-4.295e+04	1.43e+04	-3.010	0.003	-7.09e+04	-1.5e+04
numOfHighSchools	-7.805e+04	1.48e+04	-5.272	0.000	-1.07e+05	-4.9e+04
avgSchoolDistance	-7597.4985	3192.136	-2.380	0.017	-1.39e+04	-1340.354
avgSchoolRating	4.026e+04	3466.355	11.613	0.000	3.35e+04	4.7e+04
avgSchoolSize	-50.3377	14.994	-3.357	0.001	-79.728	-20.948
MedianStudentsPerTeacher	-2090.5051	3369.968	-0.620	0.535	-8696.232	4515.222
numOfBathrooms	1.216e+05	5208.575	23.348	0.000	1.11e+05	1.32e+05
numOfBedrooms	-7.013e+04	5313.602	-13.199	0.000	-8.05e+04	-5.97e+04
numOfStories	-1.065e+05	7209.404	-14.771	0.000	-1.21e+05	-9.24e+04
Dmy_association_True	-1.805e+05	8491.534	-21.259	0.000	-1.97e+05	-1.64e+05
Dmy_cooling_True	-2.83e+04	3.22e+04	-0.879	0.379	-9.14e+04	3.48e+04
Dmy_garage_True	-4.647e+04	1.3e+04	-3.567	0.000	-7.2e+04	-2.09e+04
Dmy_heating_True	8.566e+04	4.43e+04	1.933	0.053	-1183.104	1.72e+05
Dmy_spa_True	2.254e+04	1.15e+04	1.958	0.050	-23.165	4.51e+04
Dmy_view_True	1.272e+04	7474.462	1.702	0.089	-1931.769	2.74e+04

From this first model and without going into more detail we can see that it is not very good, we can affirm that it is significant ("F-statistic" very high and "Prob (F-statistic)" equal to 0) but the R-squared is only 0.528 value which is not very good. In addition, there are many non-significant variables as we can see from the p-values.

For example "numOfSecurityFeatures", which is surprising, but it may be because Austin is considered a safe city by the majority. It is also surprising that the dummy "dmy_cooling_True" is also not relevant since Austin's climate is warm as it is close to Mexico. However, we see why this can be the case and that is that in the vast majority of our data homes have cooling systems, which is why the regressor estimates that in the most common cases this characteristic is covered:

```
Dmy_cooling_True
1    13876
0      220
dtype: int64
```



At the same time, it is surprising how little importance "lotSizeSqFt" is that from the beginning we assumed to be relevant but should not be so. However, we are reassured to see that this can be explained by "livingAreaSqFt" (this is significant) which has a similar meaning and can provide similar and sufficient information. The other variables that we discard may make more logical sense in our view.

So the first thing we will do before drawing more conclusions will be to dispense with the non-significant variables: "parkingSpaces", "numOfPhoto", "numOfAccessibilityFeatures", "MedianStudentsPerTeacher", "Dmy_cooling_True", "lotSizeSqFt", "numOfAppliances", "Dmy_view_True", "numOfPatioAndPorchFeatures", "numOfSecurityFeatures", "Dmy_heating_True".

With this we obtain a model without these variables and we see how we do not lose R-squared, that is, it does not explain our model less even if we have removed a few variables, which supports that they were not important. The model is now simpler while maintaining its previous goodness.

OLS Regression Results

Dep. Variable:	latestPrice	R-squared:	0.528
Model:	OLS	Adj. R-squared:	0.527
Method:	Least Squares	F-statistic:	449.1
Date:	Wed, 03 Jan 2024	Prob (F-statistic):	0.00
Time:	11:53:48	Log-Likelihood:	-1.5861e+05
No. Observations:	11276	AIC:	3.173e+05
Df Residuals:	11247	BIC:	3.175e+05
Df Model:	28		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	3.696e+08	1.91e+07	19.360	0.000	3.32e+08	4.07e+08
zipcode	-4425.5509	213.612	-20.718	0.000	-4844.268	-4006.834
latitude	-3.425e+05	4.86e+04	-7.050	0.000	-4.38e+05	-2.47e+05
longitude	7.846e+05	6.46e+04	12.148	0.000	6.58e+05	9.11e+05
propertyTaxRate	-1.025e+06	7.26e+04	-14.106	0.000	-1.17e+06	-8.82e+05
yearBuilt	-845.4483	197.650	-4.277	0.000	-1232.877	-458.019
numPriceChanges	-1.237e+04	1215.165	-10.181	0.000	-1.48e+04	-9989.079
latest_salemonth	2688.8464	953.273	2.821	0.005	820.264	4557.428
latest_saleyear	3.443e+04	4105.285	8.386	0.000	2.64e+04	4.25e+04
numOfParkingFeatures	3.882e+04	6254.047	6.207	0.000	2.66e+04	5.11e+04
numOfWaterfrontFeatures	6.012e+05	4.57e+04	13.164	0.000	5.12e+05	6.91e+05
numOfWindowFeatures	-1.541e+04	6321.113	-2.437	0.015	-2.78e+04	-3016.272
numOfCommunityFeatures	-7.1e+04	2.49e+04	-2.852	0.004	-1.2e+05	-2.22e+04
livingAreaSqFt	261.0676	5.313	49.134	0.000	250.652	271.483
numOfPrimarySchools	9.51e+04	2.73e+04	3.481	0.001	4.15e+04	1.49e+05
numOfElementarySchools	1.477e+05	2.41e+04	6.130	0.000	1e+05	1.95e+05
numOfMiddleSchools	-4.084e+04	1.38e+04	-2.955	0.003	-6.79e+04	-1.37e+04
numOfHighSchools	-7.476e+04	1.43e+04	-5.221	0.000	-1.03e+05	-4.67e+04
avgSchoolDistance	-7679.9739	3181.979	-2.414	0.016	-1.39e+04	-1442.739
avgSchoolRating	3.922e+04	3084.769	12.713	0.000	3.32e+04	4.53e+04
avgSchoolSize	-53.6346	14.006	-3.829	0.000	-81.089	-26.180
numOfBathrooms	1.218e+05	5197.728	23.431	0.000	1.12e+05	1.32e+05
numOfBedrooms	-7.01e+04	5304.044	-13.217	0.000	-8.05e+04	-5.97e+04
numOfStories	-1.055e+05	7179.580	-14.691	0.000	-1.2e+05	-9.14e+04
Dmy_association_True	-1.805e+05	8481.838	-21.281	0.000	-1.97e+05	-1.64e+05
Dmy_garage_True	-4.26e+04	9777.998	-4.356	0.000	-6.18e+04	-2.34e+04
Dmy_spa_True	2.513e+04	1.13e+04	2.224	0.026	2981.322	4.73e+04

Next, we evaluated collinearity with FIV.

	feature	VIF
0	const	4.524646e+07
1	zipcode	1.273806e+00
2	latitude	2.653416e+00
3	longitude	3.486896e+00
4	propertyTaxRate	1.831043e+00
5	yearBuilt	2.113432e+00
6	numPriceChanges	1.061203e+00
7	latest_salemonth	1.118522e+00
8	latest_saleyear	1.724963e+00
9	numOfAppliances	1.122471e+00
10	numOfParkingFeatures	2.769255e+00
11	numOfPatioAndPorchFeatures	1.861713e+00
12	numOfSecurityFeatures	1.570011e+00
13	numOfWaterfrontFeatures	1.010743e+00
14	numOfWindowFeatures	1.279459e+00
15	numOfCommunityFeatures	1.035697e+00
16	livingAreaSqFt	3.718186e+00
17	numOfPrimarySchools	4.780546e+00
18	numOfElementarySchools	3.644830e+00
19	numOfMiddleSchools	1.561960e+00
20	numOfHighSchools	2.139708e+00
21	avgSchoolDistance	1.336187e+00
22	avgSchoolRating	3.825826e+00
23	avgSchoolSize	2.421404e+00
24	numOfBathrooms	3.402276e+00
25	numOfBedrooms	2.009249e+00
26	numOfStories	1.617032e+00
27	Dmy_association_True	2.092597e+00
28	Dmy_garage_True	2.762879e+00
29	Dmy_heating_True	1.034613e+00
30	Dmy_spa_True	1.131170e+00
31	Dmy_view_True	1.128888e+00

Probably due to the fact that several variables have already been eliminated, no notable collinearity appears (there are no VIFs greater than 5).

Finally, we try to see if we can improve our model by introducing interactions between variables. We tried with eliminated variables ("dmy_heating_True" and "dmy_cooling_True") and they are not significant, with variables that we think could improve the model because they are somewhat related (between variables of the different levels of schools) and there is no improvement either. Seeing this, we thought about the possibility that having several location characteristics, we were surprised that longitude and latitude did not have collinearity with zipcode for example, so we added interaction between "longitude" and "latitude" that could make sense since the location is given by these two variables. Adding this interaction leads to an increase in collinearity that, due to hierarchy, we ignore. Get:

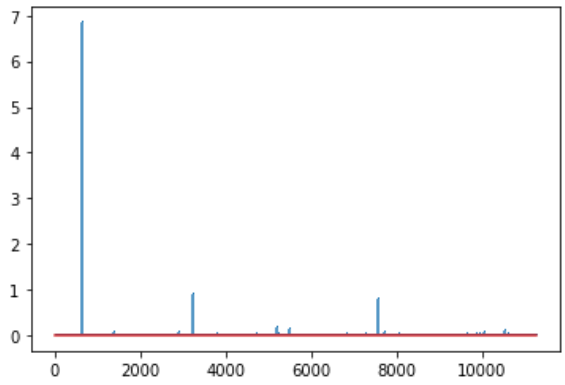
OLS Regression Results

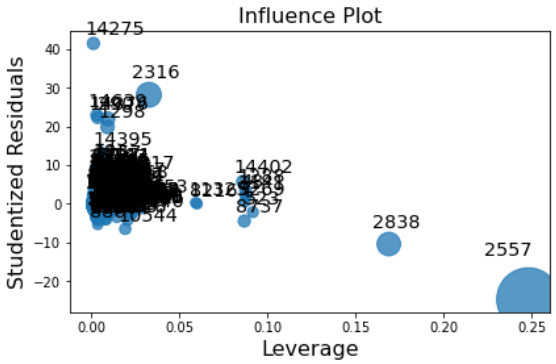
Dep. Variable:	latestPrice	R-squared:	0.531
Model:	OLS	Adj. R-squared:	0.530
Method:	Least Squares	F-statistic:	472.4
Date:	Wed, 03 Jan 2024	Prob (F-statistic):	0.00
Time:	12:56:17	Log-Likelihood:	-1.5857e+05
No. Observations:	11276	AIC:	3.172e+05
Df Residuals:	11248	BIC:	3.174e+05
Df Model:	27		
Covariance Type:	nonrobust		

lat*long	-4.573e+06	4.79e+05	-9.550	0.000	-5.51e+06	-3.63e+06

Some improvement in the explanation of the model, although subtle. The interaction is significant and improves our model, so we decided to keep it.

Looking at the results, we realize that there is a possibility that there are certain outliers that hinder our model.





As we can see, there are certain values that can distort the capacity of our model, but we will continue with them since the model worsens considerably without them.

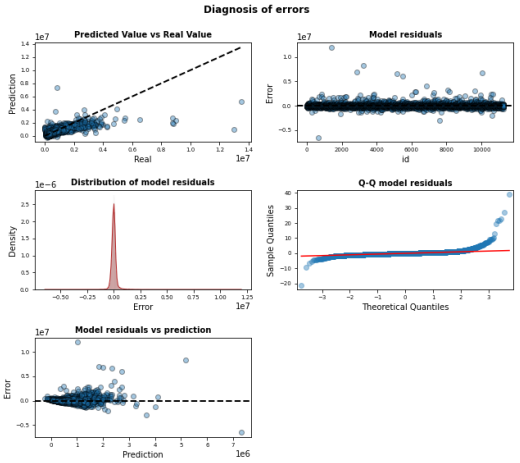
Results

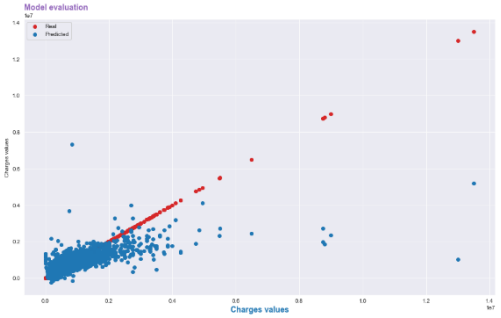
Our final model for all data is the one with the interaction with latitude and longitude:

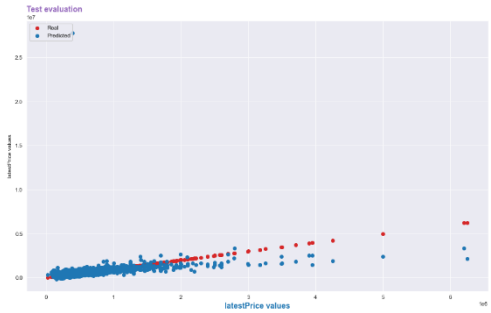
OLS Regression Results

avgSchoolSize	-79.0290	14.202	-5.565	0.000	-106.866	-51.192
numOfBathrooms	1.21e+05	5177.646	23.373	0.000	1.11e+05	1.31e+05
numOfBedrooms	-6.899e+04	5284.184	-13.056	0.000	-7.94e+04	-5.86e+04
numOfStories	-1.079e+05	7155.619	-15.085	0.000	-1.22e+05	-9.39e+04
Dmy_association_True	-1.789e+05	8449.611	-21.178	0.000	-1.96e+05	-1.62e+05
Dmy_garage_True	-4.34e+04	9739.389	-4.456	0.000	-6.25e+04	-2.43e+04
Dmy_spa_True	2.158e+04	1.13e+04	1.916	0.055	-497.639	4.37e+04
lat*long	-4.573e+06	4.79e+05	-9.550	0.000	-5.51e+06	-3.63e+06
Df Residuals:		11248	BIC:			3.174e+05
Df Model:		27				

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.386e+10	1.41e+09	9.810	0.000	1.11e+10	1.66e+10
zipcode	-3860.2058	220.645	-17.495	0.000	-4292.709	-3427.703
latitude	-4.475e+08	4.68e+07	-9.557	0.000	-5.39e+08	-3.56e+08
longitude	1.392e+08	1.45e+07	9.603	0.000	1.11e+08	1.68e+08
propertyTaxRate	-9.464e+05	7.29e+04	-12.991	0.000	-1.09e+06	-8.04e+05
yearBuilt	-697.5270	197.381	-3.534	0.000	-1084.429	-310.625
numPriceChanges	-1.25e+04	1210.092	-10.330	0.000	-1.49e+04	-1.01e+04
latest_salemonth	2688.1933	949.511	2.831	0.005	826.987	4549.400
latest_saleyear	3.471e+04	4088.803	8.490	0.000	2.67e+04	4.27e+04
numOfParkingFeatures	3.994e+04	6204.299	6.437	0.000	2.78e+04	5.21e+04
numOfWaterfrontFeatures	5.942e+05	4.55e+04	13.063	0.000	5.05e+05	6.83e+05
numOfWindowFeatures	-1.542e+04	6288.286	-2.452	0.014	-2.77e+04	-3094.483
numOfCommunityFeatures	-5.606e+04	2.48e+04	-2.257	0.024	-1.05e+05	-7369.728
livingAreaSqFt	259.3065	5.295	48.968	0.000	248.927	269.686
numOfPrimarySchools	7.428e+04	2.73e+04	2.721	0.007	2.08e+04	1.28e+05
numOfElementarySchools	1.239e+05	2.41e+04	5.137	0.000	7.66e+04	1.71e+05
numOfMiddleSchools	-4.518e+04	1.38e+04	-3.280	0.001	-7.22e+04	-1.82e+04
numOfHighSchools	-7.469e+04	1.43e+04	-5.236	0.000	-1.03e+05	-4.67e+04
avgSchoolDistance	-5024.4827	3181.471	-1.579	0.114	-1.13e+04	1211.757
avgSchoolRating	4.147e+04	3081.517	13.457	0.000	3.54e+04	4.75e+04







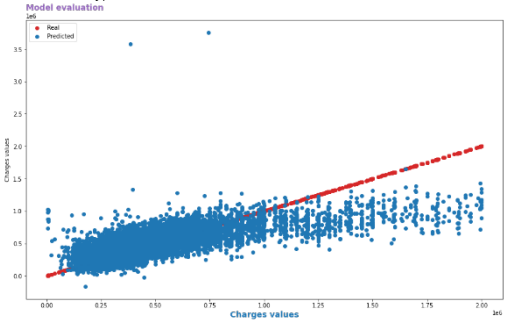

```
Root Mean Squared Error - training: 309757.75289611734
```

```
Root Mean Squared Error - test: 578380.5550499166
```

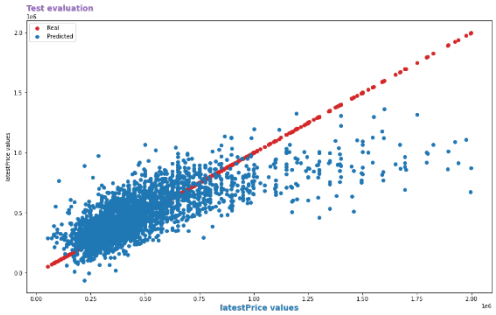
This is our final model that as we can see, is not bad for most cases up to prices of 2 million. In the test it gets much worse compared to training and we could still say that it is a bad model generalizing, but as we saw before, we get an outlier, in this case in the test, that ruins the model, so this conclusion is perhaps too drastic.
We don't quite understand how apparently this outlier spoils the model, but if the model is eliminated at a global level it gets worse. That is why we decided to make another model with reduced data in which we follow almost to the letter the previous steps to the outliers, where we will eliminate them since in this case we take the data with a price of less than 2 million (and here eliminating them does not worsen the model) to see if in this range we are able to improve the model considerably.

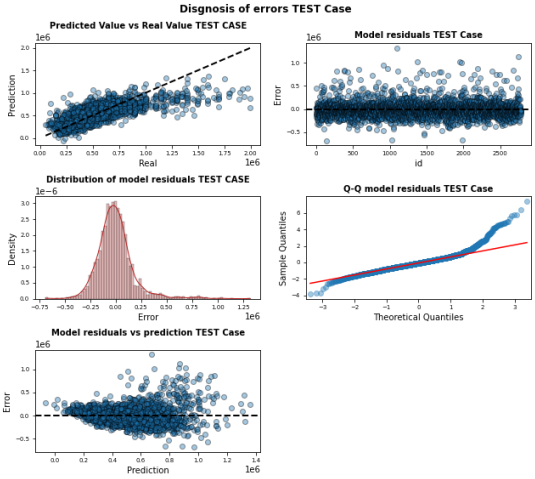
Model with reduced data

```
                            OLS Regression Results
==============================================================================
Dep. Variable:           latestPrice   R-squared:                       0.537
Model:                           OLS   Adj. R-squared:                  0.536
Method:                Least Squares   F-statistic:                     444.2
Date:               Wed, 03 Jan 2024   Prob (F-statistic):               0.00
Time:                       14:51:34   Log-Likelihood:            -1.5123e+05
No. Observations:              11133   AIC:                         3.025e+05
Df Residuals:                  11103   BIC:                         3.027e+05
Df Model:                         29
Covariance Type:            nonrobust
```




```
Root Mean Squared Error - training: 191942.0377368379
```






```
R2 Value in test: 0.56
Accuracy in test 70.94
RMSE in test:  178361.259
```

In this case with data whose price is less than 2 million we see how the RMSE has decreased a lot (also due to the elimination of outliers obviously) and although the

explanation has not increased much more, we have a much more robust model. Even so, we continue to see how from 1 million dollars, it predicts the worst values. In any case, we achieved an RMSE of less than 200,000 thousand dollars, which for a linear algorithm is not entirely a bad starting point.

Conclusions

As a first regression, the results are not very good, but with this algorithm we have been able to see certain relationships between variables, observe the existence of extreme values and ranges more suitable for the linear regression algorithm. We conclude that our algorithm is probably too simple since at the global level it does not explain the relationships between variables well enough to make a good prediction. However, it's a good starting point for the following regressors.

On the other hand, we take numerous positive aspects from this algorithm. During the stage of the importance of the variables, we learn which ones affect the price of housing to a greater and lesser extent. This is very useful since if we are going to buy or sell a house in Austin we can use this information to our advantage. For example, if you want to sell a single-family home in the city of Austin but to increase its value you want to do a previous renovation, you will know what to spend the renovation money on, which will be on the most important variables such as increasing the characteristics of the parking lot, putting a spa,... (numOfParkingFeatures, Dmy_spa_True`,...`) and not on the less important variables such as those discarded in the development steps that will not affect the price according to our model ("numOfPatioAndPorchFeatures", "numOfSecurityFeatures",...).

Decision tree, CART

Objectives

With this algorithm we want to focus above all on intrinsic information and the importance of variables. In this case, obtaining this knowledge prevails over precision, since we will then explore much more powerful algorithms in this area.

Expected results

We hope to obtain similar conclusions to the previous ones in the linear regression algorithm. Knowing what has been analyzed above, it is to be expected that variables such as "livingAreaSqFt" are the most important and others such as "numOfAccessibilityFeatures" are again not very significant.

Pre-processed

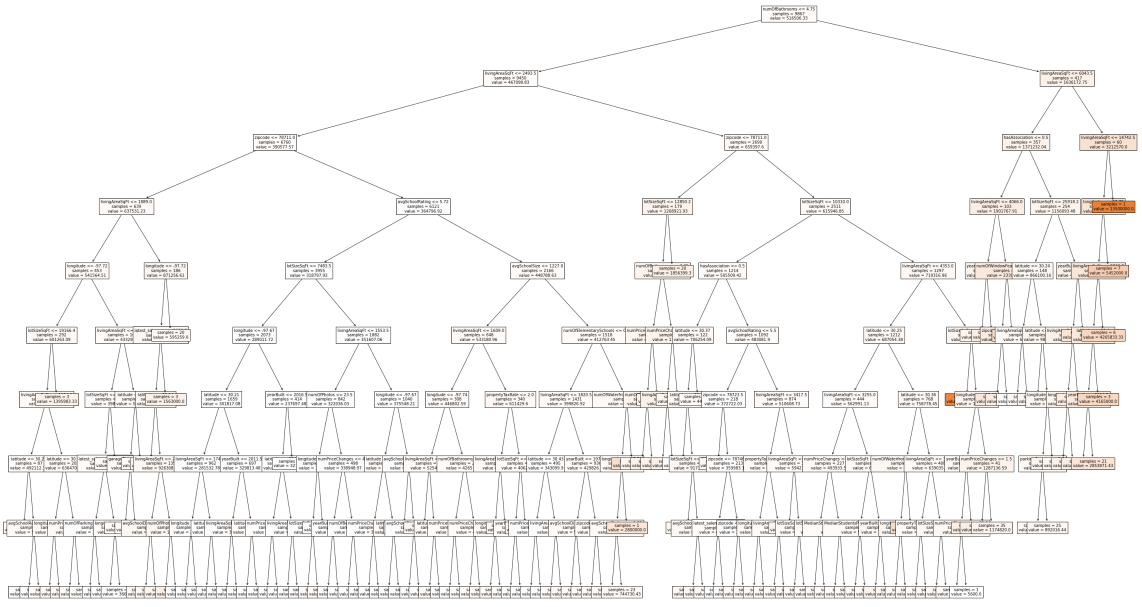
In this area it does not change much compared to the previous algorithm, we eliminate the variables that do not provide information from the beginning. In addition, we divide our dataset into train and test, we choose 80% and 20% respectively.

Development

We use GridSearch with Cross Validation to find the best tree possible. After the iterations we get the following parameters as best for a tree (at most we put 10 deep):

```
DecisionTreeRegressor(max_depth=9, min_impurity_decrease=0.0001,
                      min_samples_split=41, random_state=1)
```

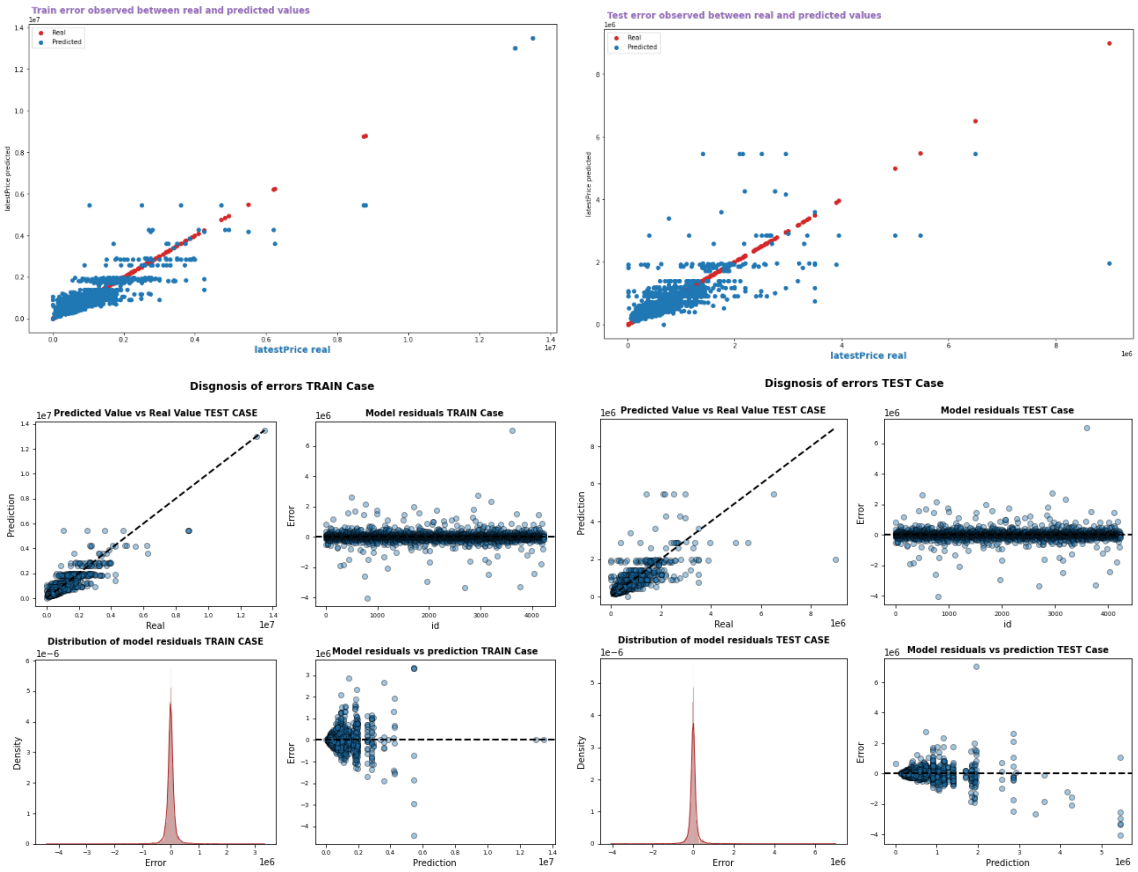
This is the entire tree, although we are only interested in the first separators as they are determined by the most important separators (we will analyze them below).



Results

Below, we will see the results obtained:

```
Root Mean Squared Error - train: 192922.34056112086
Root Mean Squared Error - test: 277000.77207850746
```



We are really surprised by the result of the decision tree, the RMSE is not as high as we expected. Admittedly, we could say that it doesn't generalize very well, probably because of outliers. However, looking at the graphs, the results are good since there is not so much dispersion compared to previous cases.

Conclusions

In this conclusion we want to focus on the importance of predictors because this is where we can obtain very valuable information. As we have highlighted before, "livingAreaSqFt" is the most important variable, which makes perfect sense. Then, as we saw in the preliminary analysis, "zipcode" follows, demonstrating the high correlation between location and price. To highlight some variable that we have not dealt with previously, we appreciate in 4th place "yearBuilt" which makes sense since the higher the year figure, the more recent it is, which makes sense but could be affected by cases in which a reform has been made, a variable that is out of our analysis as it does not have it in our dataset. On the other hand, we see that there are many variables that have not been used, hence all those that appear with importance equal to 0. In this group, "MedianStudentsPerTeacher" or "hasCooling" and

	predictor	importance			
26	livingAreaSqFt	0.593936	20	numOfPatioAndPorchFeatures	0.000215
0	zipcode	0.106738	16	numOfPhotos	0.000149
35	numOfBathrooms	0.082780	8	hasHeating	0.000000
12	yearBuilt	0.036340	36	numOfBedrooms	0.000000
1	latitude	0.033324	3	propertyTaxRate	0.000000
32	avgSchoolRating	0.028294	34	MedianStudentsPerTeacher	0.000000
5	hasAssociation	0.026387	6	hasCooling	0.000000
25	lotSizeSqFt	0.026215	7	hasGarage	0.000000
2	longitude	0.021243	30	numOfHighSchools	0.000000
33	avgSchoolSize	0.013460	29	numOfMiddleSchools	0.000000
28	numOfElementarySchools	0.007035	9	hasSpa	0.000000
22	numOfWaterfrontFeatures	0.006082	27	numOfPrimarySchools	0.000000
4	garageSpaces	0.006079	10	hasView	0.000000
13	numPriceChanges	0.004769	24	numOfCommunityFeatures	0.000000
19	numOfParkingFeatures	0.003114	23	numOfWindowFeatures	0.000000
17	numOfAccessibilityFeatures	0.001992	11	parkingSpaces	0.000000
31	avgSchoolDistance	0.001506	21	numOfSecurityFeatures	0.000000
15	latest_saleyear	0.000341	14	latest_salemonth	0.000000
			18	numOfAppliances	0.000000

"hasHeating" appear again, which we have already explained could be the reason for their lack of relevance. We see how there is some variable that was significant in linear regression that in this case is not even used, as is the case of "latest_salesmonth". That is, the multiple linear regression algorithm seemed to say that the closer to the end of the year it is usually sold at a higher price, which could go hand in hand with inflation, but the CART tree did not take it as a separator, indicating that it was not relevant enough (within our depth range).

Redes Neuronales, MLP

Objectives

In this case, we do want to focus a little more on the accuracy of the regression and we are looking for a better predictor than those obtained previously.

Expected results

We hope that this algorithm will be able to better represent the relationships of our variables in order to have a robust model with consistent accuracy.

Pre-processed

Again, we start with the initial, complete data and you eliminate the columns of data that we cannot process or that do not provide us with information. From here, we transform the numerical variables (normalizing) and categorical variables (with onehotencoder). Finally, we divided the data into train and test (80% and 20%).

Development

We look for the best arrangement of our neural network with a cross-search (`RandomizedSearchCV`). As a result, we obtain that it is best to use the following:

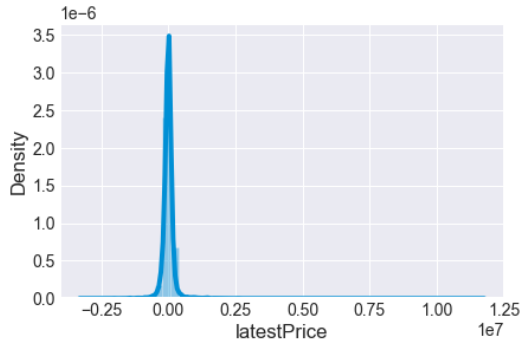
param_model__learning_rate_init	param_model__hidden_layer_sizes
0.01	10

Once this is done, we move on to evaluate our model.

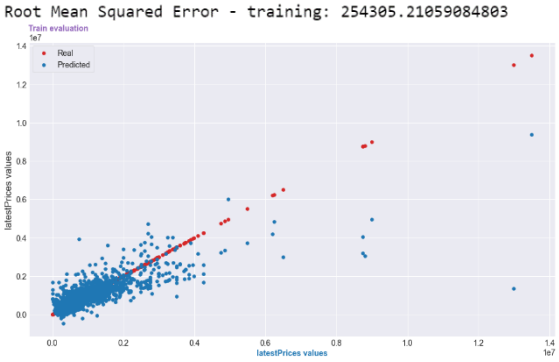
Results

We evaluate the results in the training first:

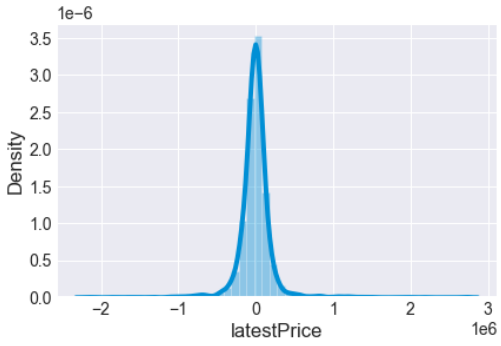
```
R_squared value in training:  0.7
Accuracy in training 67.03
RMSE in training:  254305.2106
```




```
Root Mean Squared Error - training: 254305.21059084803
```




```
R_squared value in test:  0.66
Accuracy in test 54.26
RMSE in test:  228234.6976
```




```
Root Mean Squared Error - test: 228234.69756852122
```

We see how again the extreme values worsen our results. However, both in the RMSE and in the graphs there is an improvement as a result of a better model.

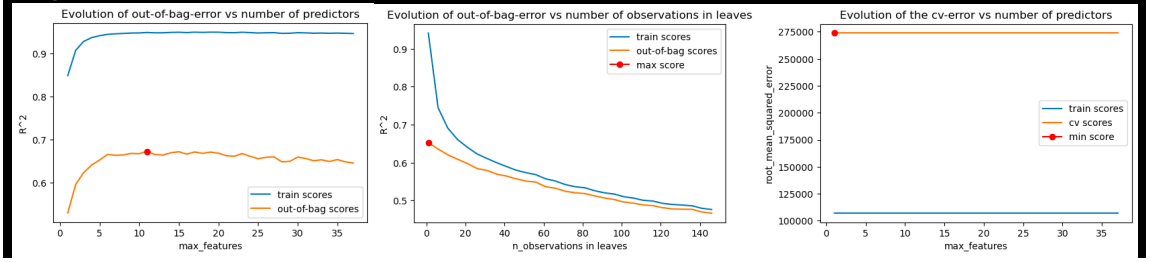
Conclusions

We cannot benefit from as many observations on our dataset as in other algorithms, but we have obtained a much more robust model that, from what we have observed between train and test, seems to generalize moderately well. Specifically, this model assumes well the existence of these extreme values that have given so much war until now. We understand that it may be due to the ability to assign weights of the neurons, which allows the characteristics of these homes with exorbitant prices to be correctly valued so that it does not damage the valuation of the other cases.

Bagging, random forest

Having already made a decision tree, you are going to make an ensemble technique that will use many simple trees at once to get a better regressor. For this, the parameters it has are very important, and that can be obtained graphically but also by executing certain code that is already looking for one.

```
Best Hyperparameters: {'max_depth': 15, 'min_samples_split': 2, 'n_estimators':
300}
```



The best result is obtained by averaging the results obtained from 300 trees, which use a random sample of the data to obtain the desired result. This should reduce overfitting and therefore have better test results than if a single decision tree had been made. To make a more robust model, the max_features parameter has been obtained by cross-validation

Results

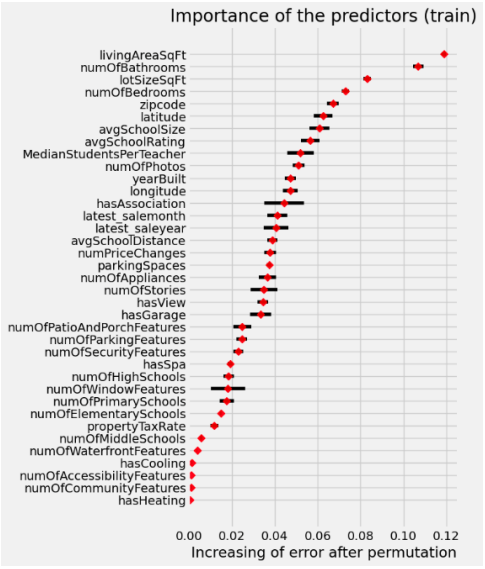
The training took him relatively well.

```
R2 Value in training: 0.85
Accuracy in training 76.01
RMSE in train: 179328.95534538347
```

The test, as expected, is worse.

```
R2 Value in test: 0.66
Accuracy in test 53.25
RMSE in test: 230575.1299296574
```

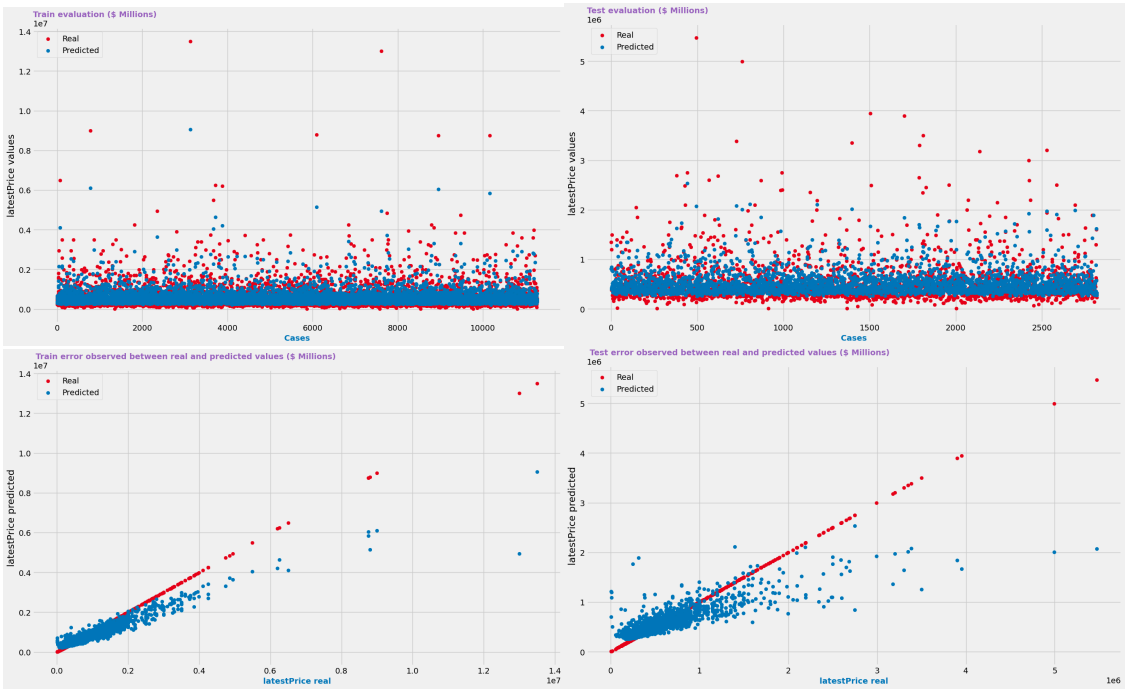
In any case, this model is worse than the previous one seen from multiple linear regression after removing the outliers, but better than before doing so. Later we will see if the same thing happens to this later.



As for the variables that are most important, there are changes here with respect to other models. The most important thing is the size of the house, followed by the bathrooms and the area, here we see a discrepancy with the first of the models (although it is not very important since it is closely related to other variables, which would be a possible explanation for its insignificance in linear regression).

On the other hand, the latitude is striking, this is because, in most cities in the United States, the good areas – and therefore expensive – tend to be found in the northern part. Then, we see variables that make a lot of sense, such as those that refer to schools, their size and their valuation. Again, we see variables such as "hasCooling" or "hasHeating" that are not important at all, so we mentioned that they are characteristics that are taken for granted as they exist in almost all the houses in the city (or at least in our dataset).

Looking graphically at the results below, it is worth repeating the results without the outliers, which are the most expensive houses. Let's assume, as in the previous model, that we remove the values of homes over $2 million.

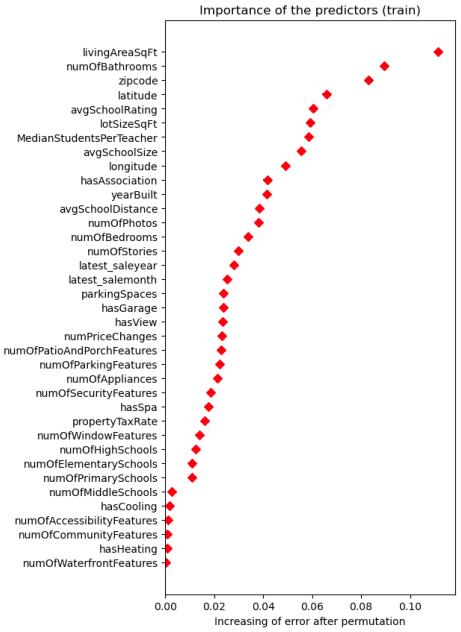


Without these outliers we find ourselves in the following situation:

```
R2 Value in training: 0.84
Accuracy in training 77.55
RMSE in train: 112036.05457281454

R2 Value in test: 0.68
Accuracy in test 64.9
RMSE in test: 157035.2742362357
```

Moderately worse in training and moderately better in test, although this can be entirely attributable to the train and test partition. What is really important is that the RMSE has been greatly lowered in both, which makes a lot of sense if you take into account that the examples that ranked the worst have been removed.



With the important variables being essentially the same. As an observation, it should be noted that these houses, being more affordable than those that have been removed, do not have waterfront features. We also see how the postal code becomes more important in this case.

Here we can see how the model would have worked even better if it had been up to the first million dollars, but due to the large number of cases that there are between 1 and 2 million, we believe that it would not be a good model if it could not take these into account.

Conclusions

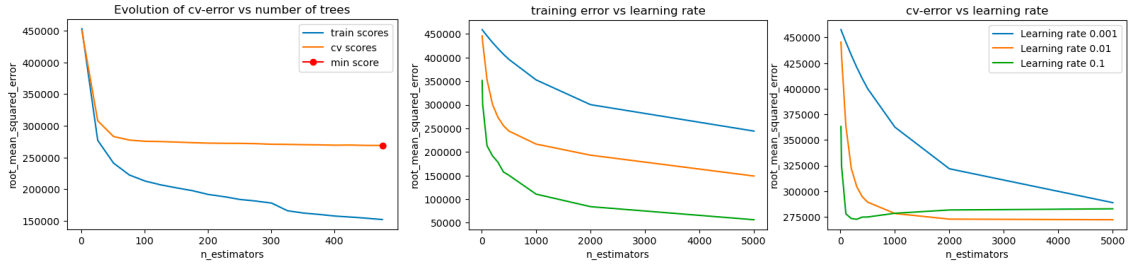
This model works better than a multiple linear regression model. As is the general trend of these models, by removing the cases that the model classifies the worst, due to the scarcity of them, the value of the RMSE has improved a lot. On the other hand, many relationships about the importance of the variables are reiterated, as we have commented, which verifies and gives more value to these conclusions drawn.

Boosting, various methods.

We will test XgBoost and LightGBM. The initial model, with all the data, and will be repeated without the outliers. Boosting starts with a simple model in which the accuracy improves.

```
R2 Value in training: 0.51
Accuracy in training 59.52
RMSE in train: 321523.1799553517
Accuracy in test 49.67
RMSE in test: 262138.53558373445
```

Poor model due to the randomness of the initial parameters. To improve it, a cross validation will be done and some better ones will be obtained.

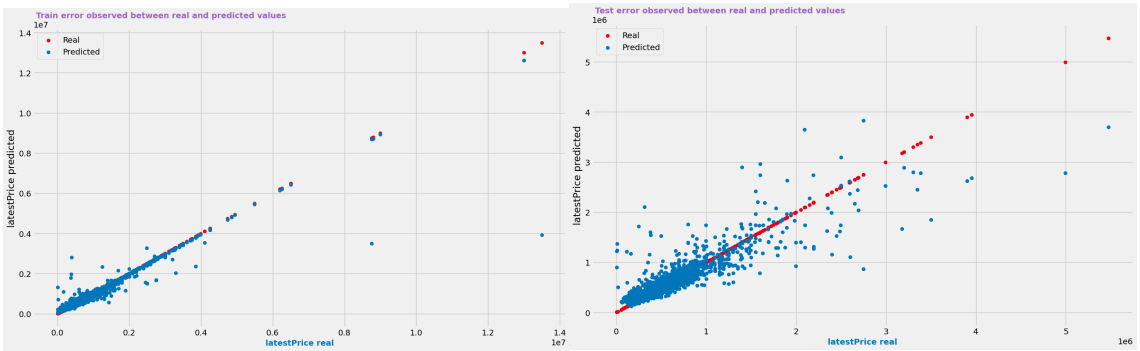



```
----------------------------------------
Best hiperparameters found (cv)
----------------------------------------
{'learning_rate': 0.01, 'max_depth': 10, 'max_features': 'sqrt', 'subsample':
1}
Number of trees of the final model: 566
```

Having obtained this for a first model, we evaluated it.

```
R2 Value in training: 0.93
Accuracy in training 89.17
RMSE in train: 123688.63831188642
Accuracy in test 58.98
RMSE in test: 185378.44210833308
```

Indeed, you can see how it has improved significantly, so much so that this first model already becomes the best (if we base ourselves on a decrease in the RMSE). The graphs of the observed error are much closer to the ideal of a model.



We continue to explore other techniques, in this case XGBoost. After performing another cross-validation to obtain the best predictor, we arrive at the following.

```
----------------------------------------
Best hiperparameters found (cv)
----------------------------------------
{'booster': 'gbtree', 'learning_rate': 0.01, 'max_depth': 10, 'subsample': 0.5}
Number of trees included in the model: 440
```

And if we now evaluate it:

```
R2 Value in training: 0.93
Accuracy in training 89.17
RMSE in train: 123688.63831188642
Accuracy in test 58.98
RMSE in test: 185378.44210833308
```

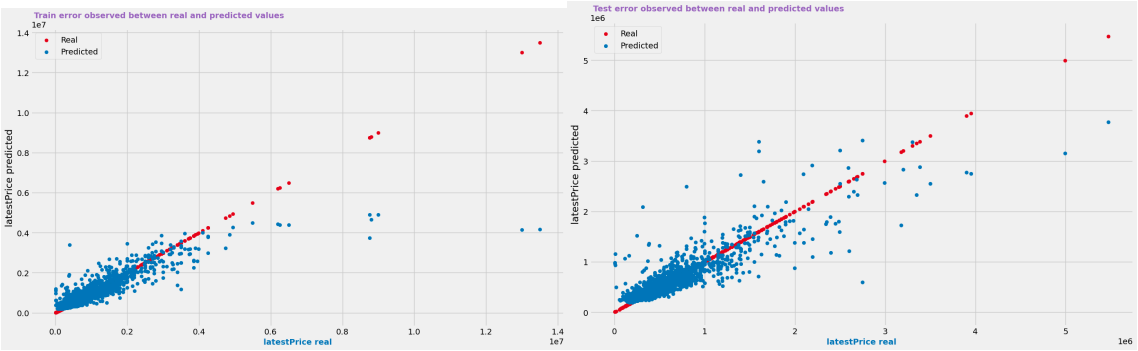
Finally, we tested LightGBM's regressor.

```
--------------------------------------
Best hiperparameters found (cv)
--------------------------------------
{'boosting_type':    'gbdt',    'learning_rate':    0.01,    'max_depth':    10,
'n_estimators': 500, 'subsample': 0.5}
R2 Value in training: 0.8
Accuracy in training 77.42
RMSE in train: 204741.79634233742
Accuracy in test 60.69
RMSE in test: 183505.11034901426
```

The accuracy is bad, but that's because of how we made the algorithm, as we get this warning:

```
[LightGBM] [Warning] Accuracy may be bad since you didn't explicitly set
num_leaves OR 2^max_depth > num_leaves. (num_leaves=31).
```

As we are evaluating the model in terms of the RMSE value in the test, this is the best model so far.



The accuracy drops with this model compared to the previous one, but it is really better for what we want.

Now we'll try the same thing by removing outliers over $2 million, which should greatly improve performance.

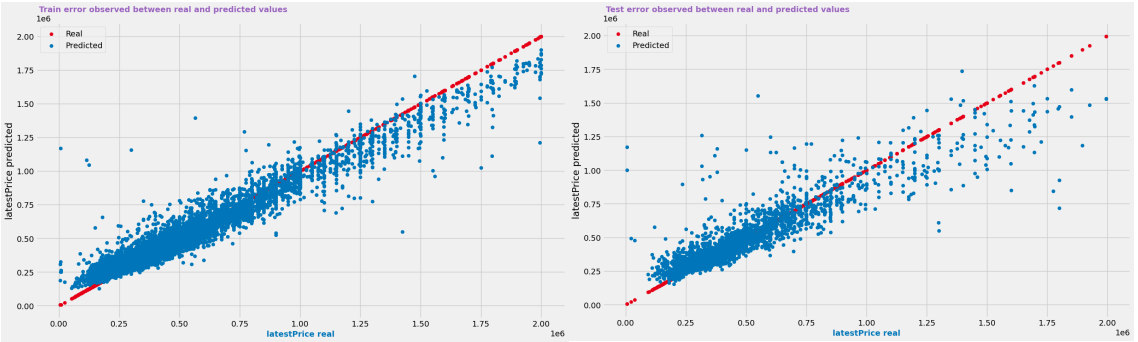
XgBoost, again the same:

```
R2 Value in training: 0.94
Accuracy in training 85.38
RMSE in train: 71087.32607158745
Accuracy in test 69.89
RMSE in test: 120918.35343189324
```

And finally LightGBM:

```
R2 Value in training: 0.9
Accuracy in training 80.27
RMSE in train: 90171.39544064907
Accuracy in test 70.87
RMSE in test: 121488.27894327733
```

In this case, XgBoost is the best model we have so far. Slightly worse accuracy in the test, but better RMSE.

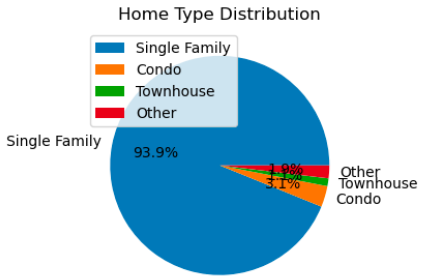


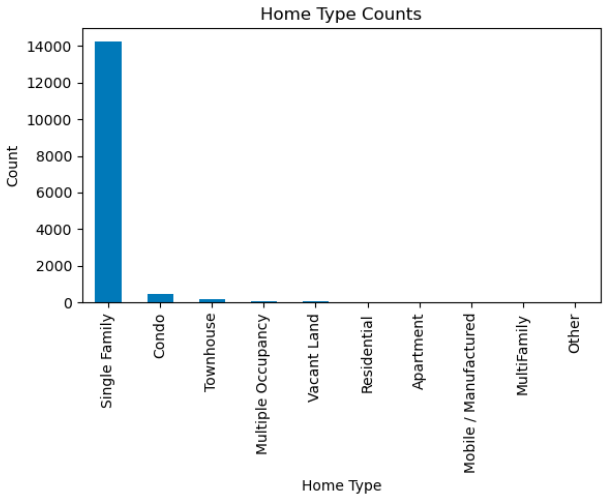
If we compare it with the previous case in which price values of more than 2 million dollars are present, it can be seen that they do fit better as the test values are closer to the line.

Concluding this section, the best model that boosting can offer is without the outliers and with XGBoost, where we get an RMSE of 121488 in the test set. This is the best model that has been obtained for regression.

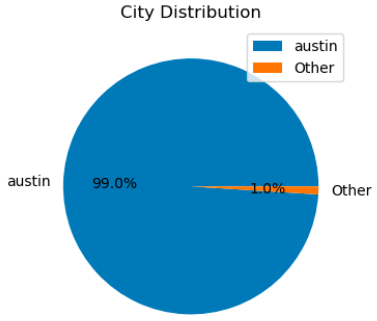
Classification

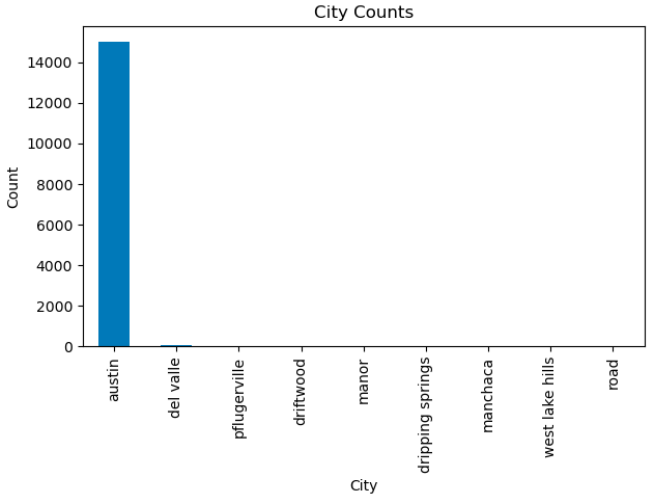
We find it interesting to use a type of data that we were not using until now, such as the type of houses that are sold. All models up to this point have consisted only of single-family homes in the city of Austin, as these are an overwhelming majority of the total cases in our database.






```
Single Family            14096
Condo 467
Townhouse                  174
Multiple Occupancy          96
Vacant Land 81
Residential                 37
Apartment                   36
Mobile / Manufactured       17
MultiFamily                 10
Other                        6
Name: homeType, dtype: int64
```





In any case, it may be interesting to make a SMOTE algorithm to obtain more cases from other types of houses, and try to make a classification. We start with these values.

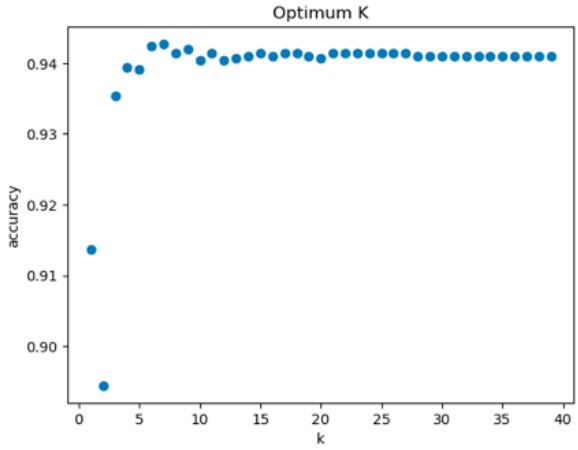
```
Single Family            14096
Condo 467
Townhouse                  174
Multiple Occupancy          96
Vacant Land 81
Residential                 37
Apartment                   36
Mobile / Manufactured       17
MultiFamily                 10
```

```
Name: homeType, dtype: int64
```

And after transforming them, we get these others:

```
Value: Apartment, Count: 10000
Value: Condo, Count: 10000
Value: Mobile / Manufactured, Count: 10000
Value: MultiFamily, Count: 10000
Value: Multiple Occupancy, Count: 10000
Value: Residential, Count: 10000
Value: Single Family, Count: 14096
Value: Townhouse, Count: 10000
Value: Vacant Land, Count: 10000
```

In principle we think that 10000 examples of each type will be enough. Having done this, we tried classifying them with a KNN model.



	precision	recall	f1-score	support
Apartment	0.00	0.00	0.00	29
Condo	0.57	0.22	0.32	374
Mobile / Manufactured	0.00	0.00	0.00	12
MultiFamily	0.00	0.00	0.00	6
Multiple Occupancy	1.00	0.01	0.03	75
Residential	0.00	0.00	0.00	31
Single Family	0.95	1.00	0.97	11270
Townhouse	0.74	0.12	0.20	145
Vacant Land	0.00	0.00	0.00	69
accuracy			0.94	12011
macro avg	0.36	0.15	0.17	12011
weighted avg	0.92	0.94	0.92	12011

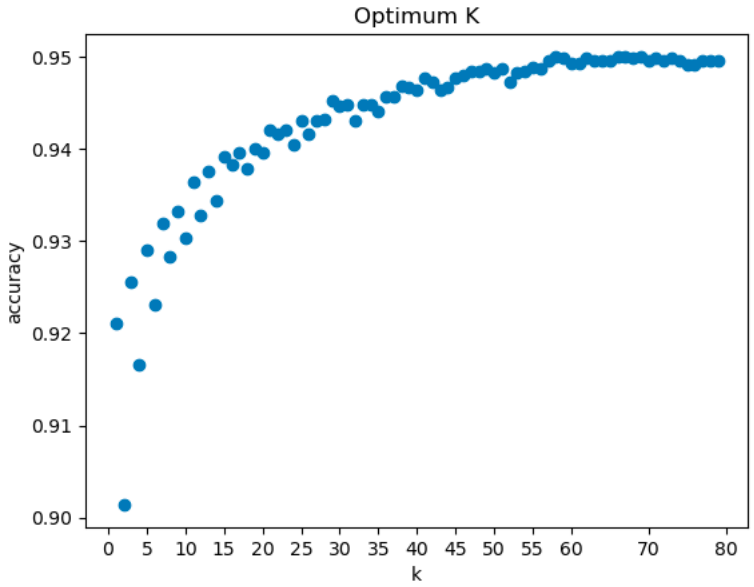
And after this first failed attempt, it is clear that of the categories where there are almost no values, it is impossible to make a good classifier. That is why from now on only the first three values are taken and evaluated from there. We have also been changing the values of the parameters, to see which one fits best. The results tend to converge towards the 95% value, this is because most of the data are single-family houses, and SMOTE is not done for the test set, so with a sufficiently large k size what will happen is that the data that are the majority will prevail. As we mentioned before, a variety of values have been tested, and the best possible model has been arrived at with this challenging database. These are the values that are going to be used for training.

```
Value: Condo, Count: 1000
Value: Single Family, Count: 9873
Value: Townhouse, Count: 1000
```

The value of single family houses has not increased, since this is the majority, while condo and townhouse houses have generated many more, to the point that single family

is double and about 8 times more in townhouse. Also, through trial and error, we find the ideal value of the k_neighbors (for the SMOTE), this being 3.
Then you have to evaluate the classifier. Seeing how sparse the data is (even after normalizing it), we decided to try distances other than Euclidean that the algorithm uses by default. After trying Manhattan (the differences are absolute), Minkowski (with different p-values) and similarity of cosines, it has been decided to choose the latter for the elaboration of the finish.
With all these factors taken into account, the final KNN model is made. This is the optimal value of k:

```
The optimal K is: 57
```



And with respect to the evaluation against the train and test sets, this is it.

```
Accuracy of K-NN classifier on training set: 0.86
[[  19  877  104]
 [   8 9803   62]
 [  11  591  398]]
              precision    recall  f1-score   support

       Condo 0.50 0.02 0.04 1000
Single Family       0.87      0.99      0.93      9873
    Townhouse       0.71      0.40      0.51      1000

     accuracy                           0.86     11873
    macro avg       0.69      0.47      0.49     11873
 weighted avg       0.82      0.86      0.82     11873

Accuracy of K-NN classifier on test set: 0.95
[[   6  131   14]
 [   1 4198   24]
 [   1   42    5]]
              precision    recall  f1-score   support

       Condo 0.75 0.04 0.08 151
Single Family       0.96      0.99      0.98      4223
    Townhouse       0.12      0.10      0.11        48

     accuracy                           0.95      4422
    macro avg       0.61      0.38      0.39      4422
 weighted avg       0.94      0.95      0.94      4422
```

It classifies the houses of a family very well, which is evident considering that they are the majority, but the precision is better than the percentage they represent, which

means that the use of the model is better than not using it, even for the least necessary, since it could be argued that it is more useful to characterize other types of houses.
As for the latter, even with the SMOTE that improves the weighted avg, he is unable to correctly distinguish the Townhouse, worse than without this process where it is the second class that has the most precision, he confuses them with single family.

With this classification system, a very high precision has been achieved and, although it is partly due to the limitations of the database - the prevalence of one type of data over others - it is satisfactory in recognizing the two major types.

Global Results

The best regression algorithm is the boosting algorithm, the LightGBM for the total data and, improving it significantly, the XGBoost for the data without outliers. As we have seen throughout the project, these outliers have been a real headache, but we have always tried to keep a version of the algorithm with them since we consider them part of what we could find when applying the model in a real case. That said, the results are satisfactory in the sense that they more than meet our initial expectations and guarantee the needs we wanted to cover.

In addition, we believe that the analysis of variables has been enriching in the sense of learning what are the most important characteristics in the real estate sector of the city of Austin. We believe that from the point of view of a home seller this information would be very useful and for a buyer, the regressor is also useful in the sense of a defense tool against scams or prices that are not in line. To sum up, the list with the most important characteristics for the price of housing will be given by the lists of importance previously taken out in the different regressors.

On the other hand, in the classification analysis we have managed to develop a model that can classify the type of housing with sufficient precision, this could be perfectly used on home buying and selling pages to filter the desired type. However, on this last point, due to the small number of cases of types other than single-family homes, we have not been able to achieve the results we would have liked.

Global Conclusions

Throughout the project we have been able to work with a set of real data of great magnitude with the difficulties that this entails. It is true that from the beginning we have imposed high expectations on the models, but, as we have been able to realize, it is really difficult to achieve high precisions on a par with the ability to generalize with a real dataset, characterized by diverse values and complex relationships.

Beyond the results obtained, there is a lot to draw useful conclusions from for any real estate company. All the variables that we have been seeing in the different regression models as the most important or the most significant should be the ones that a real estate company should improve/increase in the event of renovation to obtain greater benefits. Obviously, it would be necessary to check whether the increase in price justifies the expense.

Finally, we would like to make a comment about the usefulness of regressors. As we have mentioned initially, the database presents the sale prices for which the homes were sold and not their value. This is an important nuance because they seem to be the same concepts, but they are not. In other words, there are many external factors that we are not able to quantify, such as fads or environmental accidents. If, for example, a house was sold for 1 million dollars, it may happen that long-term works begin on that street that, for reasons such as noise, cleanliness or others, cause the price to be devalued for

that period of time since no one would want to buy that house at that specific time. Therefore, our models would fail locally in these cases.

Tools

Python 3.8.8 (VSCode)
Power BI Desktop