Politechnika
Wrocławska

Data Mining

“Exploring the US real estate market in King County”

Monday Group, Students:

Alessandro Boldorini 240207
Mathurin Gamichon 240762
Shahad,Muhammad Ali 243409

Academic year: 2017/2018

Table of Contents

Data Description

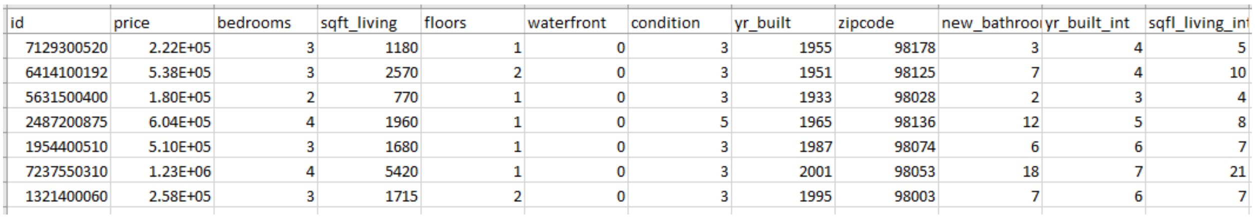
The dataset for this project we have selected it's from Kaggle (https://www.kaggle.com/bala7123/king-county/data) and the data is about King County 's (US) homes sold between May 2014 and May 2015. Below it is shown a presentation of the data set. For each house, we have a unique Identification Number, the Price on the market and some information, in particular:

- Dimension in square feet, number of bedrooms, bathrooms and floors;
- The year of construction;
- Waterfront, so if the house is built in front a source of water;
- The condition of the house (1 for the worst condition, 5 for the best);
- Zipcode, which identify the city in which the house is built.

In the last two columns of the table we can see an example of how we grouped the houses for year of construction and for dimension, in order to make some analysis easier. In particular:

- Year_built interval is [1;10], and we grouped our houses in intervals of 10 years each;
- For sqft_living interval is [1;51], and we grouped our houses in intervals of 260 sqft (so approximately 25 square meters).

The total number of observation is 21613.



id	price	bedrooms	sqft_living	floors	waterfront	condition	yr_built	zipcode	new_bathroo	yr_built_int	sqfl_living_int
7129300520	2.22E+05	3	1180	1	0	3	1955	98178	3	4	5
6414100192	5.38E+05	3	2570	2	0	3	1951	98125	7	4	10
5631500400	1.80E+05	2	770	1	0	3	1933	98028	2	3	4
2487200875	6.04E+05	4	1960	1	0	5	1965	98136	12	5	8
1954400510	5.10E+05	3	1680	1	0	3	1987	98074	6	6	7
7237550310	1.23E+06	4	5420	1	0	3	2001	98053	18	7	21
1321400060	2.58E+05	3	1715	2	0	3	1995	98003	7	6	7

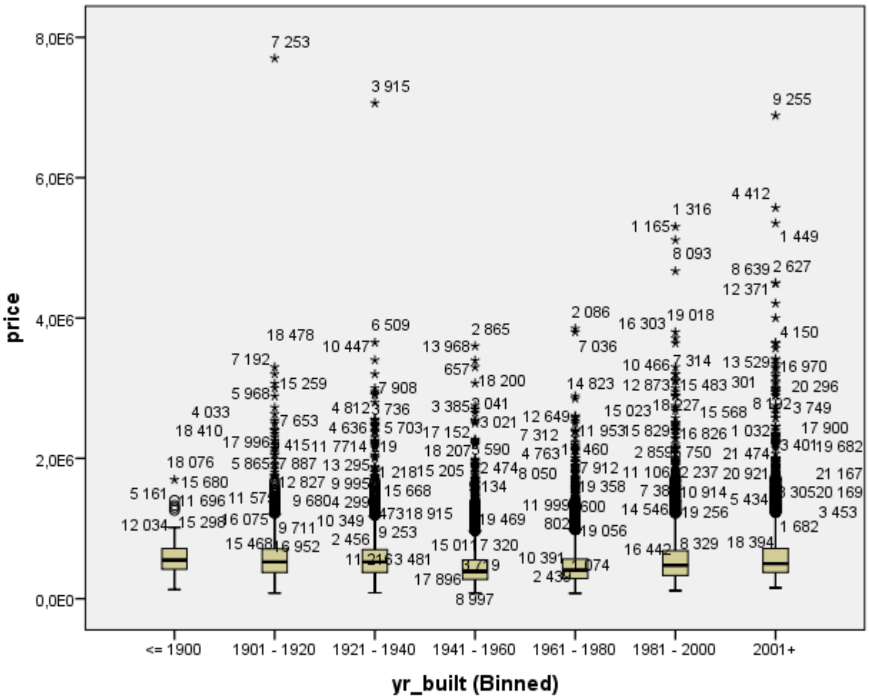
Below we show some basic statistic about our dataset:

	N	Range	Minimum	Maximum	Mean
	Statistic	Statistic	Statistic	Statistic	Statistic
price	21613	7,625 E+006	7,5 E+004	7,7 E+006	5,4009 E+005
bedrooms	21613	33	0	33	3,37
bathrooms	21613	58,00	,00	58,00	3,1148
sqft_living	21613	13250	290	13540	2079,90
floors	21613	2,5	1,0	3,5	1,494
waterfront	21613	1	0	1	,01
condition	21613	4	1	5	3,41
yr_built	21613	115	1900	2015	1971,01
zipcode	21613	198	98001	98199	98077,94
Valid N (listwise)	21613				

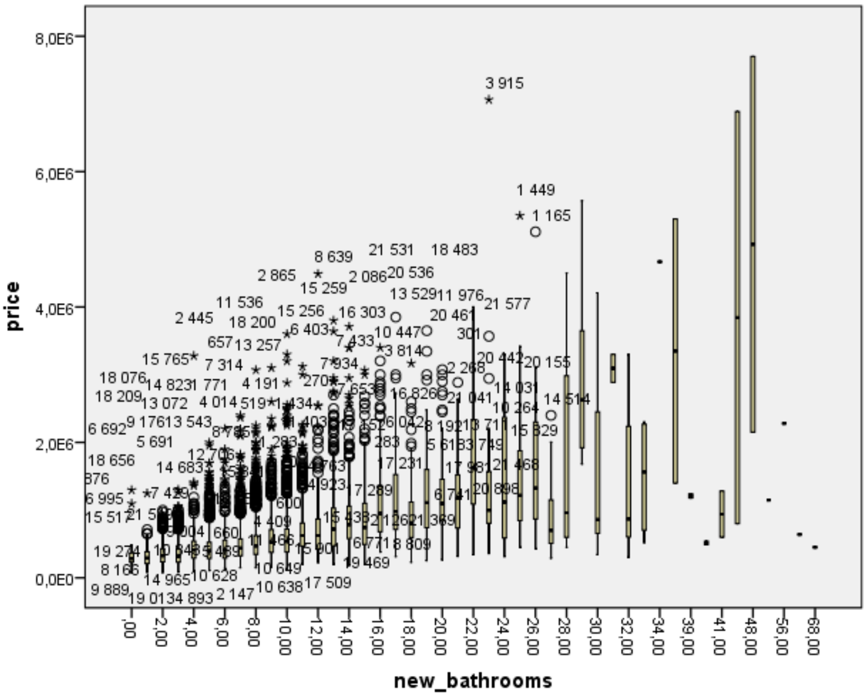
Outliers

After the basic analysis, we tried to find the outliers of every independent variable by the dependent variable (price).
We realized that there are so much outliers. This is probably because our data set is huge, more than 21 thousand houses but also because the house market isn’t standardized. For example, there are some extra big houses or “little castles” which can’t really be compared with a “normal” house. Also, we think that the most expensive houses (which are just a few number of observation) are automatically considered ad outliers, because of the high price.
We provide two examples of boxplots, showing this problem about the outliers:

Independent variable: yr_build.



Independent variable new_bathrooms:



As we can see there is a lot of outliers in absolute value but considering the percentage on the total number of observation it remains decent.

We can also note that there are only above price outliers and not below. So, we can say that the outliers are only too expensive houses and not too cheap houses.

For the variable *new_bathrooms* some number are only for one house (41,56...) so it can't show outlier for these values.

For every other variable, the outlier graphic is similar to the two above.

We tried to find the most important outliers, that means the houses which are considered outlier in every variable. We find these 6 outliers:

ID	Price	Bedrooms	N_Bath	Sqft_living	Floors	Condition	yt_build	zipcode	waterfront
1316	$5 300 000,00	6	36	7390	2	4	1991	98040	1
1449	$5 350 000,00	5	25	8000	2	3	2009	98004	0
3915	$7 060 000,00	5	23	3320	2	3	1940	98004	1
4412	$5 570 000,00	5	28,75	10040	2	3	2001	98039	0
7253	$7 700 000,00	6	48	12050	2,5	4	1910	98102	0
9255	$6 885 000,00	6	46,5	9890	2	3	2001	98039	0

Considering the mean ($ 540.008 $) and standard deviation ($ 367.127) of the price we can see that those houses are very expensive compared to the dataset.

K-Nearest Neighbour

We applied the KNN algorithm to our data in order to make a prediction about the price of a house we choose. The process was subdivided in 3 parts:

- Analyse the data we have in order to find the best K based on our factors and dataset;
- Forecast prices of all the houses using KNN method, calculate the average mean absolute error;
- Forecast the price the house we choose.

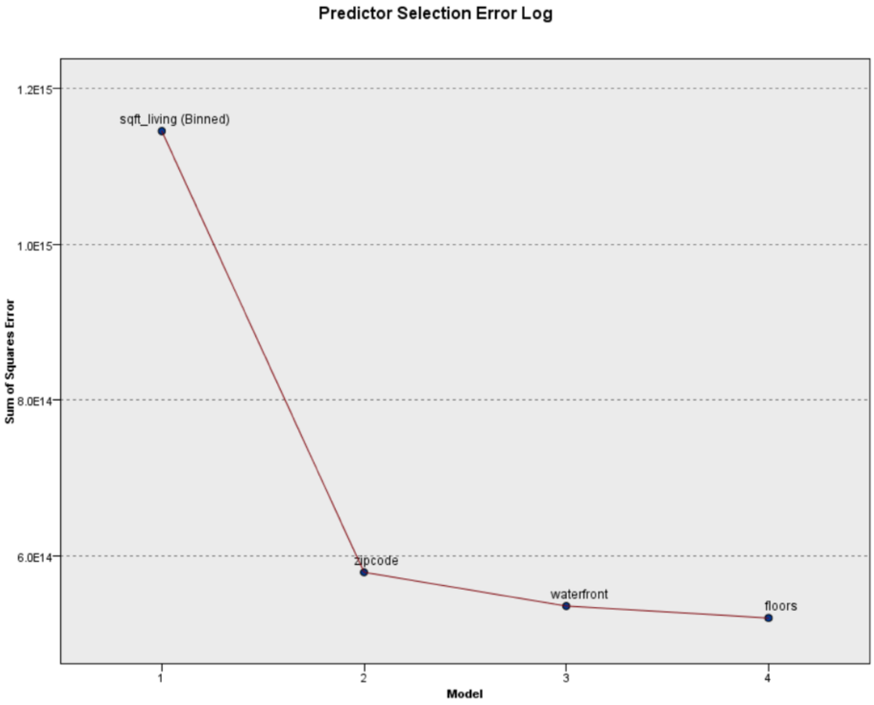
First of all, we ran the algorithm introducing some intervals in order to understand which values of the K were the most interesting. We started by introducing:

- [2;5];
- [6;10];

We saw that the most interesting values of K was around the middle of the interval [4,5,6,7], so we decided to run another time the algorithm with only these 4 values. We found that K=6 was the best result.

After that we tried to run the algorithm considering normalisation and weight of the factors. The aim of this analysis was to discover the minimum SSE and which factors were the most important to consider for the forecast. The results were:

K	Normalisation	Weighted	Feature List	SSE
K = 6	Y	N	sqft_living zipcode waterfront bedrooms bathrooms floors	5,39E+14
	Y	Y	sqft_living zipcode waterfront bathrooms bedrooms	5,29E+14
	N	Y	sqft_living zipcode waterfront	5,93E+14
	N	N	sqft_living zipcode waterfornt floors	5,20E+14



Discovered that, we proceeded in calculate the errors of our prediction.

SSE	2,43661E+32
% variation abs	18,7%

As we can see the prediction error is within an interval between [10%;20%] so we can say that KNN is a quite suitable algorithm for prediction.
After we tried to predict the price of a house. We found another house and we considered just the most important factors of KNN initial analysis. We applied the algorithm and the result we found is:

Real Price	Predicted Price	Sqft_living	Zipcode	Waterfront	Floors
6,70E+05	7,24E+05	2820	98034	NO	2

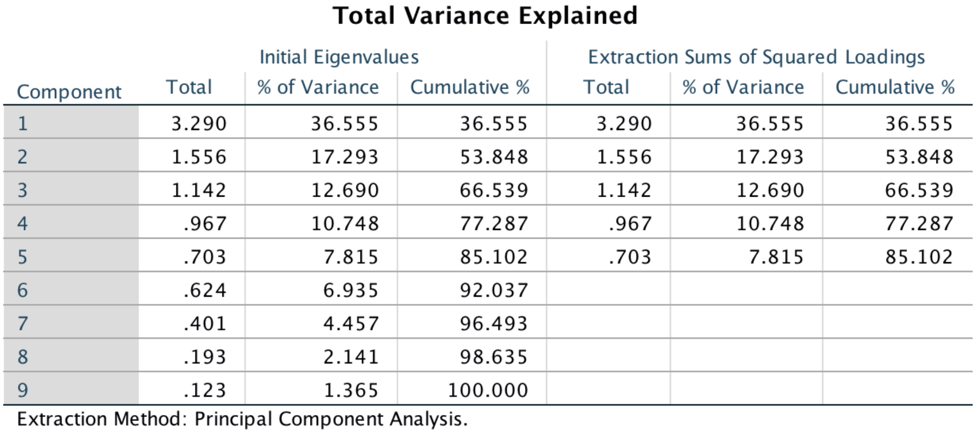
We calculate the percentage mean absolute error also for the forecasting:

% average absolute error	8,06%

The error of the prediction is good.

K means

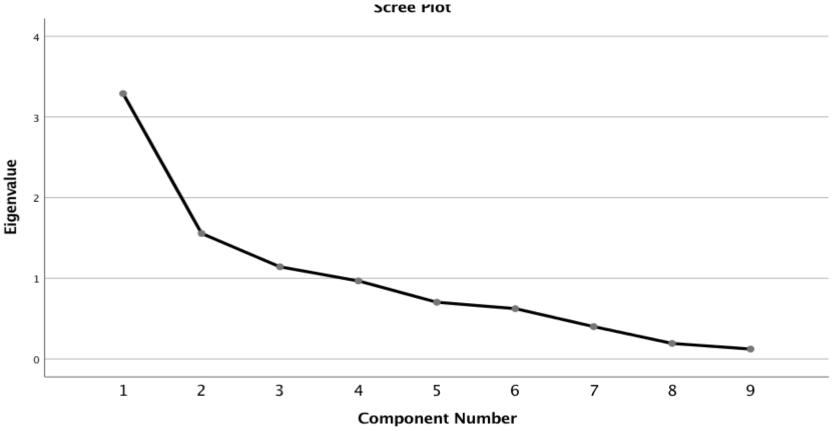
Before the implementation of the algorithm, we decided to apply a ***Factor Analysis***, to find new components that could explain better our dataset. With the *factor analysis*, we found 9 new factors.
In the table below we can see our results, the Total variance, % of Variance and Cumulative.



Total Variance Explained

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	3.290	36.555	36.555	3.290	36.555	36.555
2	1.556	17.293	53.848	1.556	17.293	53.848
3	1.142	12.690	66.539	1.142	12.690	66.539
4	.967	10.748	77.287	.967	10.748	77.287
5	.703	7.815	85.102	.703	7.815	85.102
6	.624	6.935	92.037			
7	.401	4.457	96.493			
8	.193	2.141	98.635			
9	.123	1.365	100.000			

Extraction Method: Principal Component Analysis.



Considering the result we obtained, we decided to adopt as new factors for the K-means analysis just the first 3 component. This because:

- they explain approximately the 70% of the variance;
- The eigenvalue of this components is over 1, while the other factors have an eigenvalue under 1.

Below we can see the component matrix. We highlight the most important factors correlated to our components.

Component Matrix

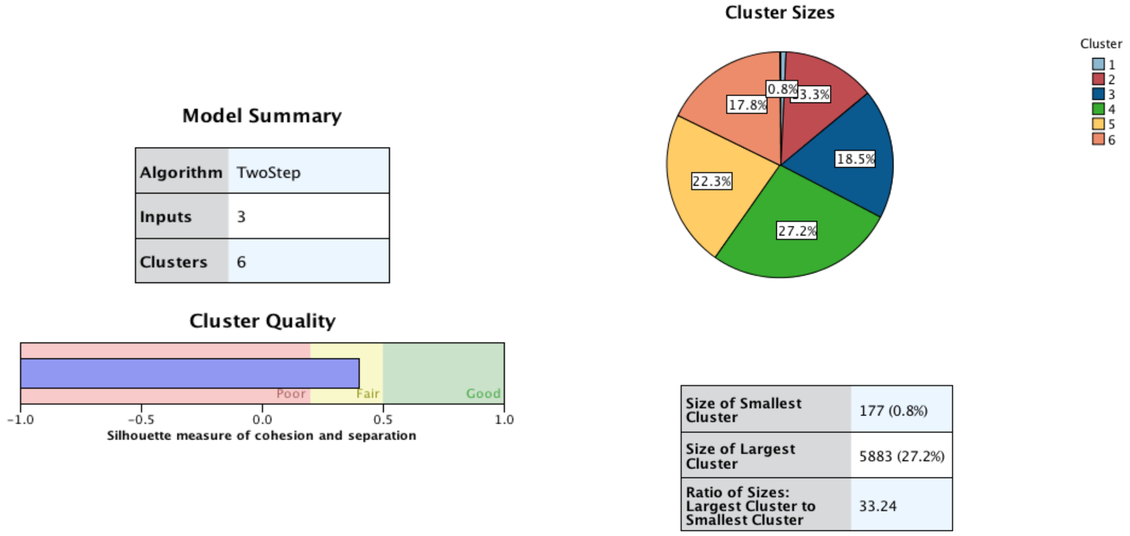
	1	2	3
price	.665	.411	.322
bedrooms	.720	.266	-.349
floors	.563	-.444	.283
waterfront	.126	.290	.695
condition	-.178	.666	-.332
zipcode	-.303	.251	.469
new_bathrooms	.910	.128	-.153
yr_built (Binned)	.515	-.686	-.006
sqft_living (Binned)	.883	.204	.017

Considering these new components, we can say that:

- Component 1, is more correlated with *bedrooms*, *bathrooms* and the dimension of the house (*sqft_living*). So, a higher value of this factor will mean that a particular house is big and with a high number of bathrooms and bedrooms.
- Component 2, is correlated with condition and *year_built*, so new houses in good conditions.
- Component 3 in correlated with *waterfront*, so if the house is in front a river, lake, seaside. We can also assume that *zipcode* is quite correlated to this factor.

The second step of the implementation of the K-means algorithm, was trying to find the best value for the K. To do this, because our dataset is quite large (more than 21000 observation), we applied the *2Step Cluster Analysis*.

The best result we had was K=6. Considering the *silhouette analysis*, as we can see in the picture below, the quality of our model is not “good”, but we decided to use this K because in the observations we had by running the algorithm a few times, this was the best result.



Model Summary

Algorithm	TwoStep
Inputs	3
Clusters	6

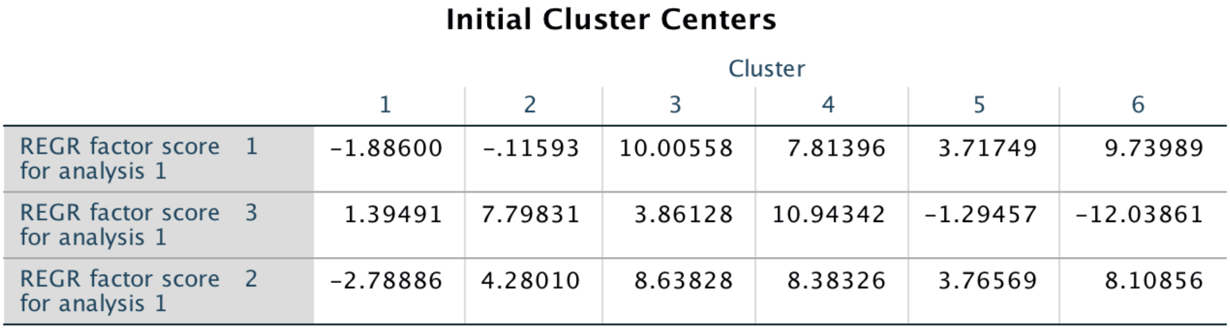
Cluster Quality

Size of Smallest Cluster	177 (0.8%)
Size of Largest Cluster	5883 (27.2%)
Ratio of Sizes: Largest Cluster to Smallest Cluster	33.24

In the table below we can have more insight about the cluster of the 2Step Cluster Analysis.

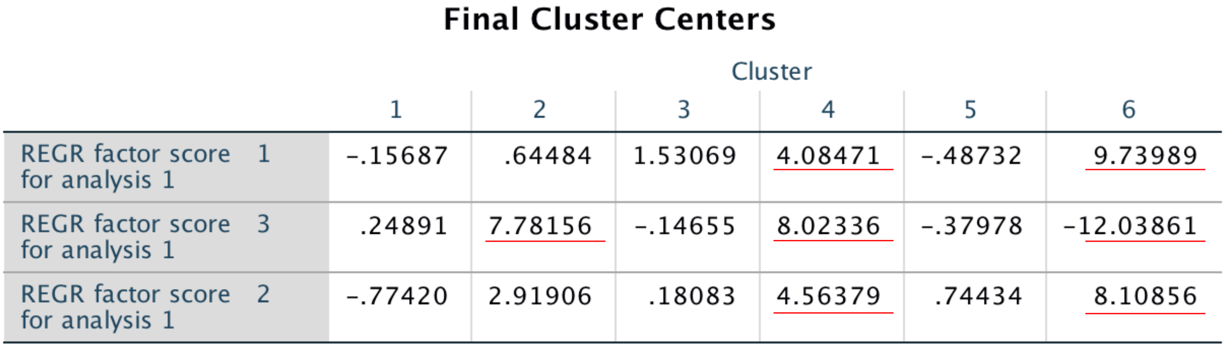
Cluster	percent(values)	percent(values)	V4	V5
1	0.819	177	1	0.818951
2	13.3253	2,880	2	13.325313
3	18.4657	3,991	3	18.465738
4	27.2197	5,883	4	27.219728
5	22.3292	4,826	5	22.329153
6	17.8411	3,856	6	17.841114

After these first two initial steps, we run the K-Means algorithm, using the new factors found through the factor analysis, and using the K of the 2step Cluster analysis. Below we can see some results taken from SPSS output of the algorithm.



Initial Cluster Centers

	Cluster					
	1	2	3	4	5	6
REGR factor score 1 for analysis 1	-1.88600	-.11593	10.00558	7.81396	3.71749	9.73989
REGR factor score 3 for analysis 1	1.39491	7.79831	3.86128	10.94342	-1.29457	-12.03861
REGR factor score 2 for analysis 1	-2.78886	4.28010	8.63828	8.38326	3.76569	8.10856



Final Cluster Centers

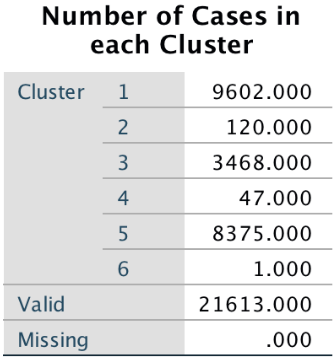
	Cluster					
	1	2	3	4	5	6
REGR factor score 1 for analysis 1	-.15687	.64484	1.53069	4.08471	-.48732	9.73989
REGR factor score 3 for analysis 1	.24891	7.78156	-.14655	8.02336	-.37978	-12.03861
REGR factor score 2 for analysis 1	-.77420	2.91906	.18083	4.56379	.74434	8.10856

To better understand how a cluster works we give now some examples of description of these clusters, considering the components we had in the 1st step (Factor analysis, Component Matrix table). In particular:

- Cluster 4 and Cluster 6, compared to the other, have the highest value for the first factor. So, we can say that these clusters contain big houses, with a huge amount of bathroom bedrooms.

- Cluster 2, Cluster 4 and Cluster 6 have the highest value of factor 2. So, considering the component of factor 2, these clusters contain the newer houses in good conditions.

For each cluster, the number of observations is shown in the table be



Number of Cases in each Cluster

Cluster	1	9602.000
	2	120.000
	3	3468.000
	4	47.000
	5	8375.000
	6	1.000
Valid		21613.000
Missing		.000

Cluster 1 is the one with the most number observations.

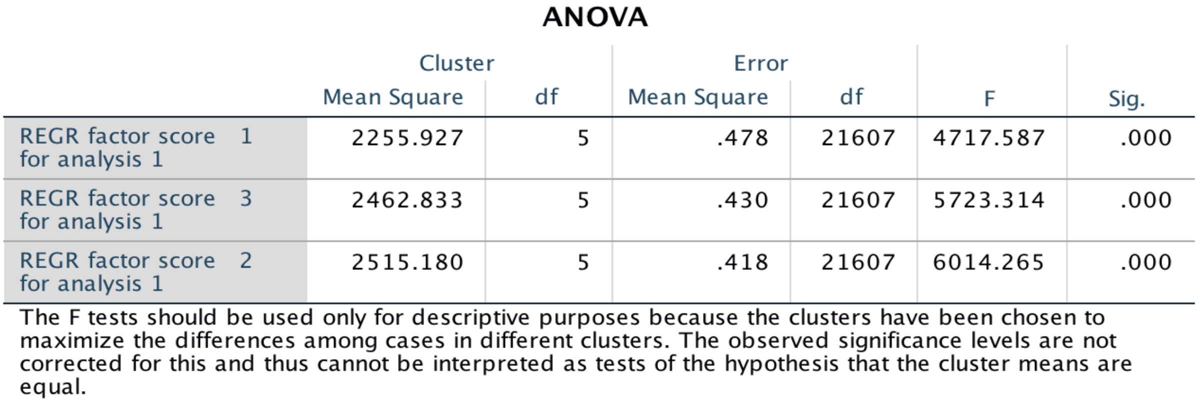
Cluster 6 instead, has just one observation. We found this result quite interesting, so we deepened our analysis about this. In particular, for Cluster 6 the observation is the n° 15871. In the tables below we present in detail the characteristic of this observation and the values for each factors.

Price	Bedroom	Sqft_liv	Floors	Waterfront	Condition	Year	Zipcode	Bathrooms
6.4E+005	33	1620 (group n°7)	1.0	0	5	1947 (group n° 4)	98103	58.00

Factor 1	Factor 2	Factor 3
9.73989	8.10856	-12.0386

This house has the highest number of Bedrooms and Bathrooms, and is in the best condition on the market (we remember that 5 is the maximum grade). We think that those are the reason than stand behind the fact this is the only observation of cluster 6.

Considering the Anova table, looking to the column "significance", because we don't have over the 5%, we can say that our factors explain very well the model.



ANOVA

	Cluster		Error			
	Mean Square	df	Mean Square	df	F	Sig.
REGR factor score 1 for analysis 1	2255.927	5	.478	21607	4717.587	.000
REGR factor score 3 for analysis 1	2462.833	5	.430	21607	5723.314	.000
REGR factor score 2 for analysis 1	2515.180	5	.418	21607	6014.265	.000

The F tests should be used only for descriptive purposes because the clusters have been chosen to maximize the differences among cases in different clusters. The observed significance levels are not corrected for this and thus cannot be interpreted as tests of the hypothesis that the cluster means are equal.

We calculate also the criterion function of our results. Because the value is quite high, we can say that our results are good.

$$CF=\frac{BCV}{WCV}=350.883$$

In conclusion of our algorithm implementation we tried to forecast the price of the house we choose for the KNN analysis. We know that the K-means is not a forecasting algorithm. We obtained this result:

Price	Cluster 2, avg price	Sqft_living	Zipcode	Waterfront	Floors	% error
6,70E+05	6,55E+04	2820	98034	0	2	-42%

The house is part of cluster number 2. For the prediction, we considered as “predicted price” the average price of the houses contained in cluster number 2. As we can see the % error in high, as we can expect from an algorithm not fitted for forecasting.

Decision tree

We tried to predict the house price with a Decision tree models. We tested 3 different tree models, CHAID, Exhaustive CHAID and CRT.

Exhaustive Chaid method:

Specifications	Growing Method	EXHAUSTIVE CHAID
	Dependent Variable	price
	Independent Variables	bedrooms, floors, condition, waterfront, zipcode, sqft_living, new_bathrooms, yr_built
	Validation	None
	Maximum Tree Depth	3
	Minimum Cases in Parent Node	1000
	Minimum Cases in Child Node	500
Results	Independent Variables Included	sqft_living, zipcode, new_bathrooms
	Number of Nodes	29
	Number of Terminal Nodes	22
	Depth	3

The global error is:

Risk	
Estimate	Std. Error
73847794327,194	3839862011,159
Growing Method: EXHAUSTIVE CHAID Dependent Variable: price	

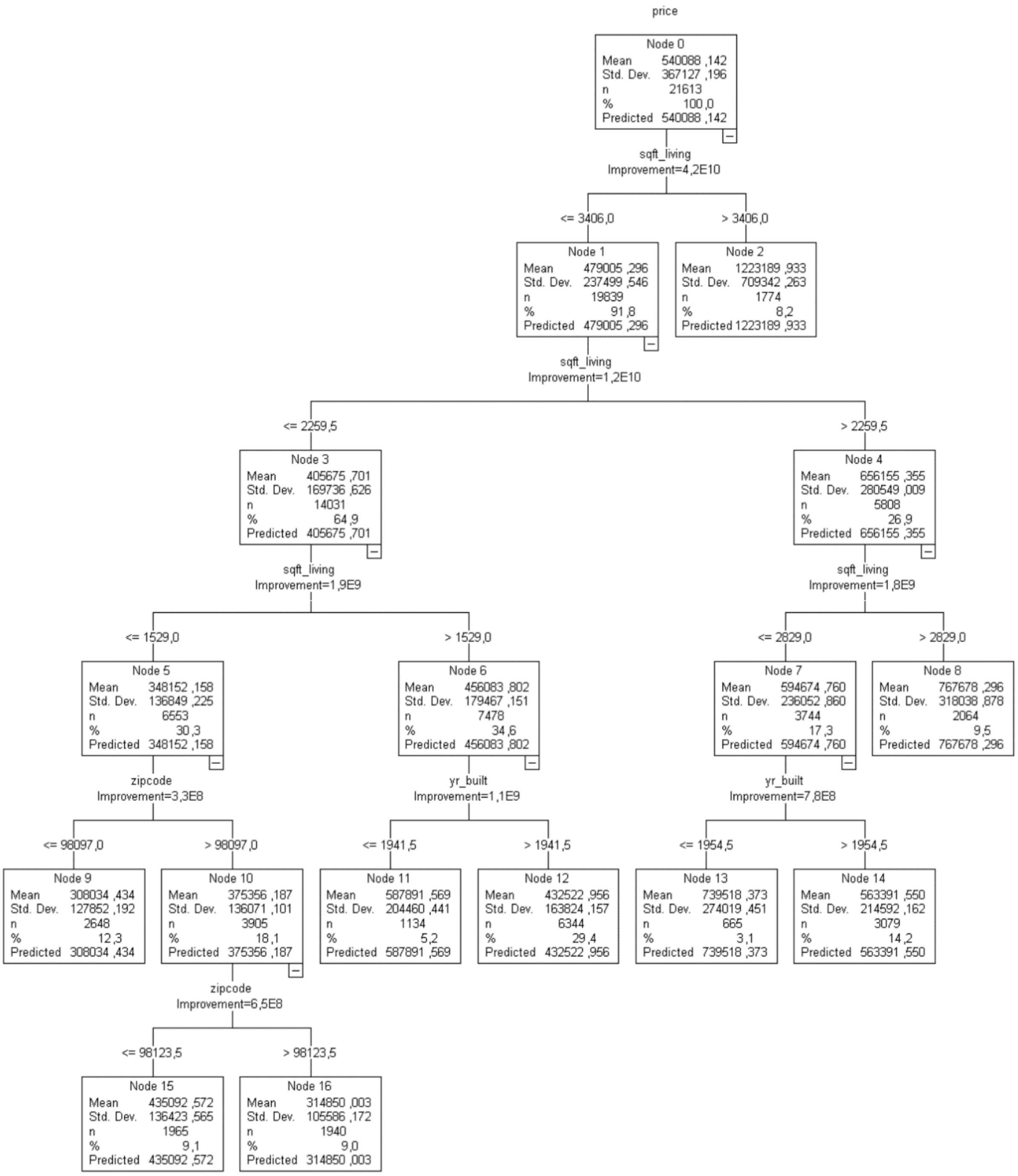
CRT method:

Specifications	Growing Method	CRT
	Dependent Variable	price
	Independent Variables	bedrooms, floors, waterfront, condition, zipcode, sqft_living, yr_built, new_bathrooms
	Validation	None
	Maximum Tree Depth	5
	Minimum Cases in Parent Node	1000
	Minimum Cases in Child Node	500
Results	Independent Variables Included	sqft_living, new_bathrooms, bedrooms, yr_built, waterfront, zipcode, floors, condition
	Number of Nodes	17
	Number of Terminal Nodes	9
	Depth	5

The global error is:

Risk	
Estimate	Std. Error
74557922610,014	3880769490,488
Growing Method: CRT Dependent Variable: price	

As example, below is shown the CRT tree:


price
Node 0
Mean 540088 ,142
Std. Dev. 367127 ,196
n 21613
% 100 ,0
Predicted 540088 ,142
sqft_living
Improvement=4,2E10
<= 3406,0
> 3406,0
Node 1
Mean 479005 ,296
Std. Dev. 237499 ,546
n 19839
% 91 ,8
Predicted 479005 ,296
Node 2
Mean 1223189 ,933
Std. Dev. 709342 ,263
n 1774
% 8 ,2
Predicted 1223189 ,933
sqft_living
Improvement=1,2E10
<= 2259,5
> 2259,5
Node 3
Mean 405675 ,701
Std. Dev. 169736 ,626
n 14031
% 64 ,9
Predicted 405675 ,701
Node 4
Mean 656155 ,355
Std. Dev. 280549 ,009
n 5808
% 26 ,9
Predicted 656155 ,355
sqft_living
Improvement=1,9E9
sqft_living
Improvement=1,8E9
<= 1529,0
> 1529,0
<= 2829,0
> 2829,0
Node 5
Mean 348152 ,158
Std. Dev. 136849 ,225
n 6553
% 30 ,3
Predicted 348152 ,158
Node 6
Mean 456083 ,802
Std. Dev. 179467 ,151
n 7478
% 34 ,6
Predicted 456083 ,802
Node 7
Mean 594674 ,760
Std. Dev. 236052 ,860
n 3744
% 17 ,3
Predicted 594674 ,760
Node 8
Mean 767678 ,296
Std. Dev. 318038 ,878
n 2064
% 9 ,5
Predicted 767678 ,296
zipcode
Improvement=3,3E8
yr_built
Improvement=1,1E9
yr_built
Improvement=7,8E8
<= 98097,0
> 98097,0
<= 1941,5
> 1941,5
<= 1954,5
> 1954,5
Node 9
Mean 308034 ,434
Std. Dev. 127852 ,192
n 2648
% 12 ,3
Predicted 308034 ,434
Node 10
Mean 375356 ,187
Std. Dev. 136071 ,101
n 3905
% 18 ,1
Predicted 375356 ,187
Node 11
Mean 587891 ,569
Std. Dev. 204460 ,441
n 1134
% 5 ,2
Predicted 587891 ,569
Node 12
Mean 432522 ,956
Std. Dev. 163824 ,157
n 6344
% 29 ,4
Predicted 432522 ,956
Node 13
Mean 739518 ,373
Std. Dev. 274019 ,451
n 665
% 3 ,1
Predicted 739518 ,373
Node 14
Mean 563391 ,550
Std. Dev. 214592 ,162
n 3079
% 14 ,2
Predicted 563391 ,550
zipcode
Improvement=6,5E8
<= 98123,5
> 98123,5
Node 15
Mean 435092 ,572
Std. Dev. 136423 ,565
n 1965
% 9 ,1
Predicted 435092 ,572
Node 16
Mean 314850 ,003
Std. Dev. 105586 ,172
n 1940
% 9 ,0
Predicted 314850 ,003

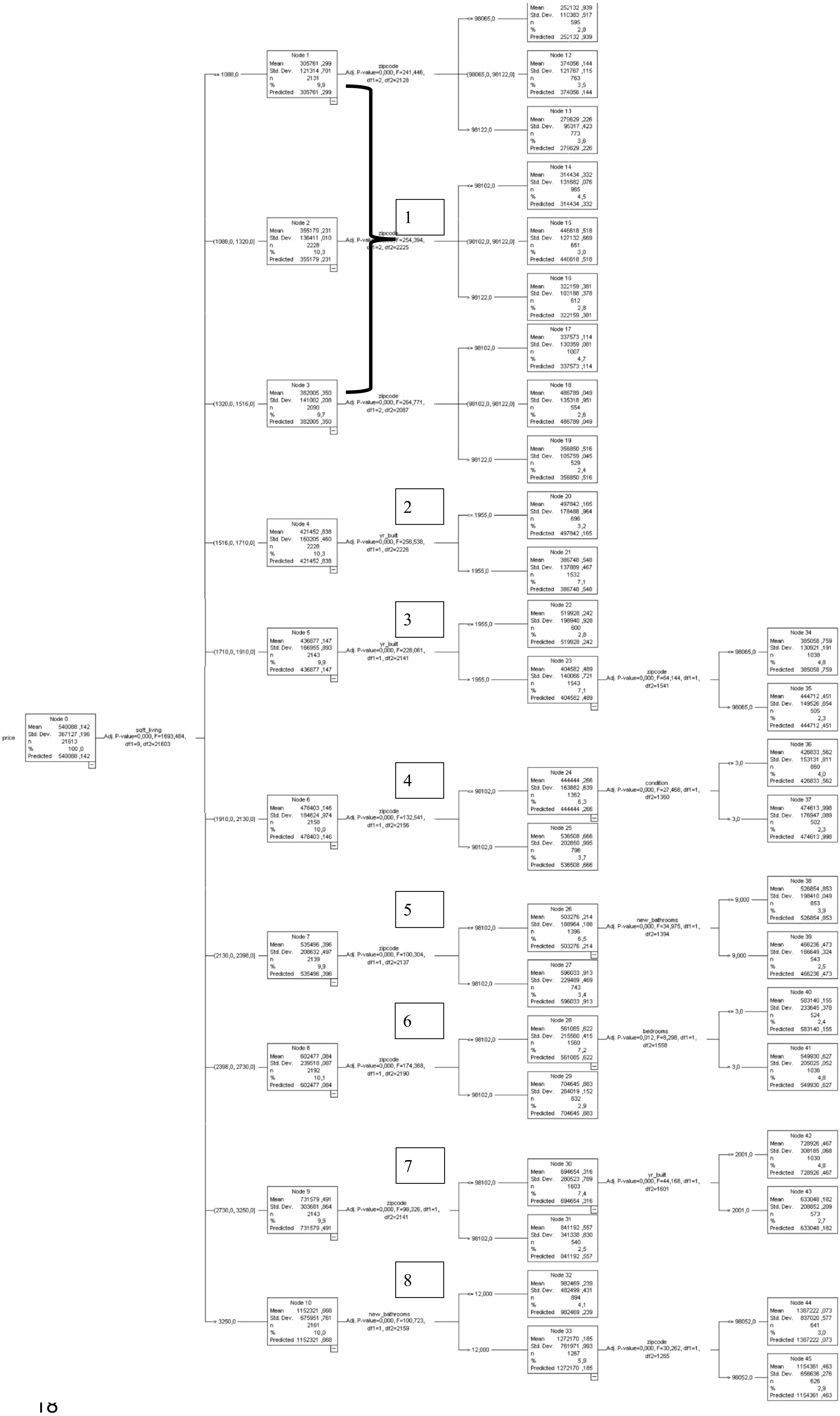
Chaid method

Specifications	Growing Method	CHAID
	Dependent Variable	price
	Independent Variables	bedrooms, floors, condition, waterfront, zipcode, sqft_living, new_bathrooms, yr_built
	Validation	None
	Maximum Tree Depth	3
	Minimum Cases in Parent Node	1000
	Minimum Cases in Child Node	500
Results	Independent Variables Included	sqft_living, zipcode, yr_built, condition, new_bathrooms, bedrooms
	Number of Nodes	46
	Number of Terminal Nodes	29
	Depth	3

The global error is:

Risk	
Estimate	Std. Error
73099843676,453	3780791491,745
Growing Method: CHAID Dependent Variable: price	

Chaid method tree is the model with the smallest error.
So, we used this tree to deepen our analysis. Below is showed the tree:


price
Node 0
Mean 540088 ,142
Std. Dev. 367127 ,196
n 21613
% 100 ,0
Predicted 540088 ,142
sqft_living
Adj. P-value=0,000, F=1693,484,
df1=9, df2=21603
<= 1088,0
(1088,0, 1320,0]
(1320,0, 1516,0]
(1516,0, 1710,0]
(1710,0, 1910,0]
(1910,0, 2130,0]
(2130,0, 2398,0]
(2398,0, 2730,0]
(2730,0, 3250,0]
> 3250,0
Node 1
Mean 305761 ,299
Std. Dev. 121314 ,701
n 2131
% 9,9
Predicted 305761 ,299
zipcode
Adj. P-value=0,000, F=241,446,
df1=2, df2=2128
<= 98065,0
(98065,0, 98122,0]
> 98122,0
Mean 252132 ,939
Std. Dev. 110383 ,517
n 595
% 2,8
Predicted 252132 ,939
Node 12
Mean 374056 ,144
Std. Dev. 121767 ,115
n 763
% 3,5
Predicted 374056 ,144
Node 13
Mean 279629 ,226
Std. Dev. 95317 ,423
n 773
% 3,6
Predicted 279629 ,226
1
Node 2
Mean 355179 ,231
Std. Dev. 136411 ,010
n 2228
% 10,3
Predicted 355179 ,231
zipcode
Adj. P-val F=254,394,
df1=2, df2=2225
<= 98102,0
(98102,0, 98122,0]
> 98122,0
Node 14
Mean 314434 ,332
Std. Dev. 131682 ,076
n 965
% 4,5
Predicted 314434 ,332
Node 15
Mean 446618 ,518
Std. Dev. 127132 ,669
n 651
% 3,0
Predicted 446618 ,518
Node 16
Mean 322159 ,381
Std. Dev. 103188 ,378
n 612
% 2,8
Predicted 322159 ,381
Node 3
Mean 382005 ,350
Std. Dev. 141002 ,208
n 2090
% 9,7
Predicted 382005 ,350
zipcode
Adj. P-value=0,000, F=264,771,
df1=2, df2=2087
<= 98102,0
(98102,0, 98122,0]
> 98122,0
Node 17
Mean 337573 ,114
Std. Dev. 130359 ,081
n 1007
% 4,7
Predicted 337573 ,114
Node 18
Mean 486789 ,049
Std. Dev. 135318 ,951
n 554
% 2,6
Predicted 486789 ,049
Node 19
Mean 356850 ,516
Std. Dev. 105759 ,045
n 529
% 2,4
Predicted 356850 ,516
2
Node 4
Mean 421452 ,838
Std. Dev. 160205 ,460
n 2228
% 10,3
Predicted 421452 ,838
yr_built
Adj. P-value=0,000, F=256,538,
df1=1, df2=2226
<= 1955,0
> 1955,0
Node 20
Mean 497842 ,165
Std. Dev. 178488 ,964
n 696
% 3,2
Predicted 497842 ,165
Node 21
Mean 386748 ,548
Std. Dev. 137889 ,467
n 1532
% 7,1
Predicted 386748 ,548
3
Node 5
Mean 436877 ,147
Std. Dev. 166955 ,893
n 2143
% 9,9
Predicted 436877 ,147
yr_built
Adj. P-value=0,000, F=228,061,
df1=1, df2=2141
<= 1955,0
> 1955,0
Node 22
Mean 519928 ,242
Std. Dev. 198940 ,928
n 600
% 2,8
Predicted 519928 ,242
Node 23
Mean 404582 ,489
Std. Dev. 140066 ,721
n 1543
% 7,1
Predicted 404582 ,489
zipcode
Adj. P-value=0,000, F=64,144, df1=1,
df2=1541
<= 98065,0
> 98065,0
Node 34
Mean 385058 ,759
Std. Dev. 130921 ,191
n 1038
% 4,8
Predicted 385058 ,759
Node 35
Mean 444712 ,451
Std. Dev. 149526 ,654
n 505
% 2,3
Predicted 444712 ,451
4
Node 6
Mean 478403 ,146
Std. Dev. 184624 ,974
n 2158
% 10,0
Predicted 478403 ,146
zipcode
Adj. P-value=0,000, F=132,541,
df1=1, df2=2156
<= 98102,0
> 98102,0
Node 24
Mean 444444 ,266
Std. Dev. 163882 ,639
n 1362
% 6,3
Predicted 444444 ,266
Node 25
Mean 536508 ,666
Std. Dev. 202850 ,995
n 796
% 3,7
Predicted 536508 ,666
condition
Adj. P-value=0,000, F=27,468, df1=1,
df2=1360
<= 3,0
> 3,0
Node 36
Mean 426833 ,562
Std. Dev. 153131 ,811
n 860
% 4,0
Predicted 426833 ,562
Node 37
Mean 474613 ,998
Std. Dev. 176947 ,089
n 502
% 2,3
Predicted 474613 ,998
5
Node 7
Mean 535496 ,396
Std. Dev. 208632 ,497
n 2139
% 9,9
Predicted 535496 ,396
zipcode
Adj. P-value=0,000, F=100,304,
df1=1, df2=2137
<= 98102,0
> 98102,0
Node 26
Mean 503276 ,214
Std. Dev. 188964 ,188
n 1396
% 6,5
Predicted 503276 ,214
Node 27
Mean 596033 ,913
Std. Dev. 229489 ,469
n 743
% 3,4
Predicted 596033 ,913
new_bathrooms
Adj. P-value=0,000, F=34,975, df1=1,
df2=1394
<= 9,000
> 9,000
Node 38
Mean 526854 ,853
Std. Dev. 198410 ,049
n 853
% 3,9
Predicted 526854 ,853
Node 39
Mean 466236 ,473
Std. Dev. 166649 ,324
n 543
% 2,5
Predicted 466236 ,473
6
Node 8
Mean 602477 ,084
Std. Dev. 239518 ,087
n 2192
% 10,1
Predicted 602477 ,084
zipcode
Adj. P-value=0,000, F=174,368,
df1=1, df2=2190
<= 98102,0
> 98102,0
Node 28
Mean 561085 ,622
Std. Dev. 215560 ,415
n 1560
% 7,2
Predicted 561085 ,622
Node 29
Mean 704645 ,883
Std. Dev. 264019 ,152
n 632
% 2,9
Predicted 704645 ,883
bedrooms
Adj. P-value=0,012, F=8,298, df1=1,
df2=1558
<= 3,0
> 3,0
Node 40
Mean 583140 ,155
Std. Dev. 233645 ,378
n 524
% 2,4
Predicted 583140 ,155
Node 41
Mean 549930 ,627
Std. Dev. 205025 ,052
n 1036
% 4,8
Predicted 549930 ,627
7
Node 9
Mean 731579 ,491
Std. Dev. 303681 ,864
n 2143
% 9,9
Predicted 731579 ,491
zipcode
Adj. P-value=0,000, F=98,326, df1=1,
df2=2141
<= 98102,0
> 98102,0
Node 30
Mean 694654 ,316
Std. Dev. 280523 ,789
n 1603
% 7,4
Predicted 694654 ,316
Node 31
Mean 841192 ,557
Std. Dev. 341338 ,830
n 540
% 2,5
Predicted 841192 ,557
yr_built
Adj. P-value=0,000, F=44,168, df1=1,
df2=1601
<= 2001,0
> 2001,0
Node 42
Mean 728926 ,467
Std. Dev. 308185 ,068
n 1030
% 4,8
Predicted 728926 ,467
Node 43
Mean 633048 ,182
Std. Dev. 208852 ,209
n 573
% 2,7
Predicted 633048 ,182
8
Node 10
Mean 1152321 ,668
Std. Dev. 675951 ,761
n 2161
% 10,0
Predicted 1152321 ,668
new_bathrooms
Adj. P-value=0,000, F=100,723,
df1=1, df2=2159
<= 12,000
> 12,000
Node 32
Mean 982469 ,239
Std. Dev. 482499 ,431
n 894
% 4,1
Predicted 982469 ,239
Node 33
Mean 1272170 ,185
Std. Dev. 761971 ,993
n 1267
% 5,9
Predicted 1272170 ,185
zipcode
Adj. P-value=0,000, F=30,262, df1=1,
df2=1265
<= 98052,0
> 98052,0
Node 44
Mean 1387222 ,073
Std. Dev. 837020 ,577
n 641
% 3,0
Predicted 1387222 ,073
Node 45
Mean 1154361 ,463
Std. Dev. 656636 ,276
n 626
% 2,9
Predicted 1154361 ,463

The decision tree works in this way (description from top to bottom, numbers of the list correlated with numbers of the picture in the previous page):

- First it checks the *sqft_living* value and divides the data set into 10 nodes:
    1. For the first 3 leaf, it checks the *zipcode* value and divides it in 9 final nodes.
    2. For the 4th leaf it checks *year_build* and divides it in 2 final nodes.
    3. For the 5th leaf it checks again *year_build* value and then divides it in 2 nodes, one is a final, and the other it checks the *zipcode* and divides it in 2 final nodes.
    4. For the 6th leaf it checks the *zipcode* value, then divides the tree in 2 nodes, one is a final, and the other it checks the *new_bathroom* and divides it in 2 final nodes.
    5. For the 7th leaf it checks the *zipcode* value, then divides it in 2 nodes, one is a final, and for the other it checks *new_bathroom* and divides it in 2 final nodes.
    6. For the 8th leaf it checks *zipcode* value then divide it in 2 nodes, one is a final, and for the other it checks the *bedrooms* and divides it in 2 final nodes.
    7. For the 9th leaf it checks *zipcode* value, then divides it in 2 nodes, one is a final, and for the other it checks the *year_built* and divides it in 2 final nodes.
    8. For the 10th leaf it checks *new_bathroom* value then divides it in 2 nodes, one is a final, and for the other it checks *zipcode* and divides it in 2 final nodes.

We also predicted the price of our house with this decision tree model.

Real price	sqft living	yr build	Node	Zipcode	Predicted Price	% abs error
6,70E+05	2820	1960	42	98034	7,29E+05	8,80%

We calculate the error between the predicted price and the real one:

% mean absolute error	% mean absolute error (prediction)
33,2%	15,91%

The error is quite huge, the algorithm is not suitable to our dataset to predict.

Conclusion

After the application of all the algorithms we calculate the absolute mean percentage error between:

- the real prices of the houses, provided by the dataset;
- the predicted prices we had from the application of the algorithm.

The results are the following:

Method	% mean absolute error	% mean error in prediction
KNN	18,7%	8,06%
Decision Tree (CHAID)	33,20%	15,91%
K-means	39,37%	41,74%

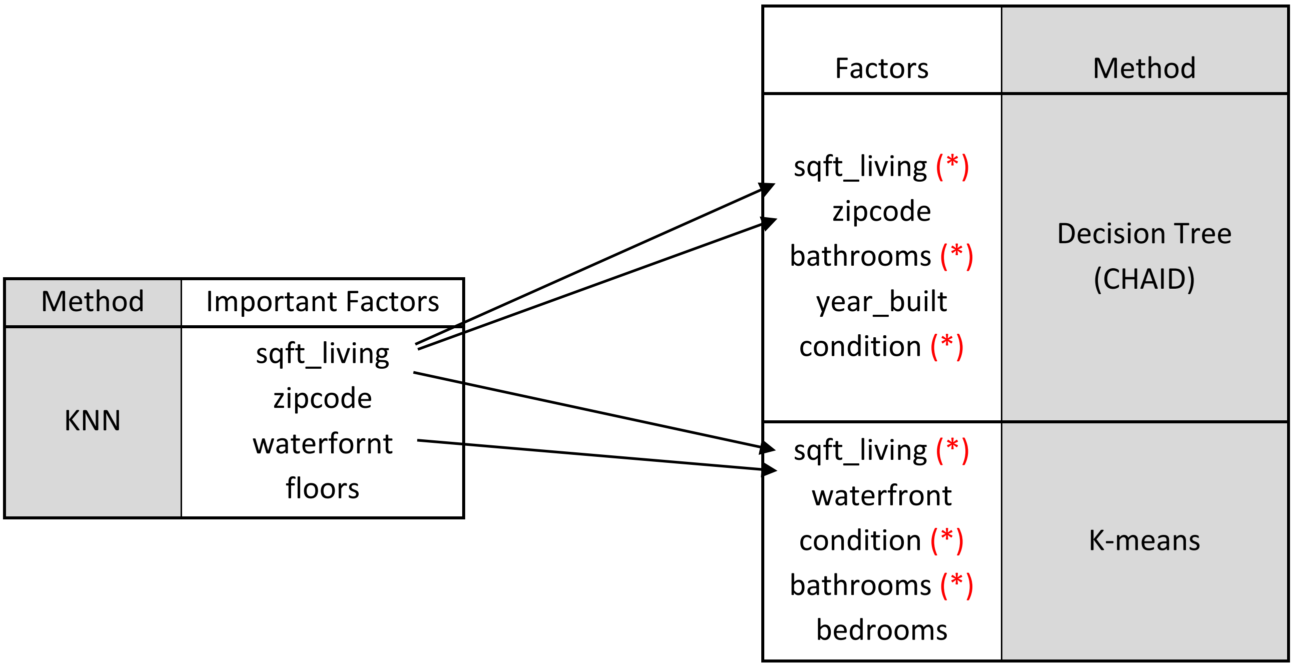
- As expected the worst error is with the K-means algorithm. This because K-means is not an algorithm for prediction. In particular, we used the average price calculated per cluster for the prediction, and this led to a high value of the error.
- The Decision Tree developed with CHAID methodology gave also us a high value of the error. As for the K-means, also in Decision Tree the prediction is based on average prices calculated for every leaf (sub-level) of the tree.
- The K-nearest Neighbour (KNN) provide us the best result in terms of % error. Because the error is in an interval between [10%;20%] we can say that KNN is a quite suitable algorithm for our dataset.

Considering our dataset and the factors we worked with, we obtained those results in terms of importance of factors in each algorithm:

Method	Factors
K-neighbours	sqft_living zipcode waterfornt floors
Decision Tree (CHAID)	sqft_living zipcode bathrooms year_built condition
K-means	sqft_living waterfront condition bathrooms bedrooms

As we can see from the table (the factors are ranked based on their importance, from top to bottom), we have some common factors and not:

- *Sqft_living* is the factors that all the 3 algorithms take into consideration, so we can say that the dimension of a house is the first factor that all the algorithms correlate to the price;
- Comparing the KNN with the other 2 algorithms we can see that they both take into account different factors:
  - Both the other two algorithms are based on a higher number of factors;
  - Both have just two of the same important factors of KNN.
  - We can assume that this difference lead to a higher error: probably in both algorithms the use of a higher number of factors lead the prediction to be affected by some disturbance made from a factor that is not very important considering the final economy of the prediction.
  - Comparing also DT and K-means we can see that they consider 2 different factors (zipcode - year built / waterfront - bedrooms).

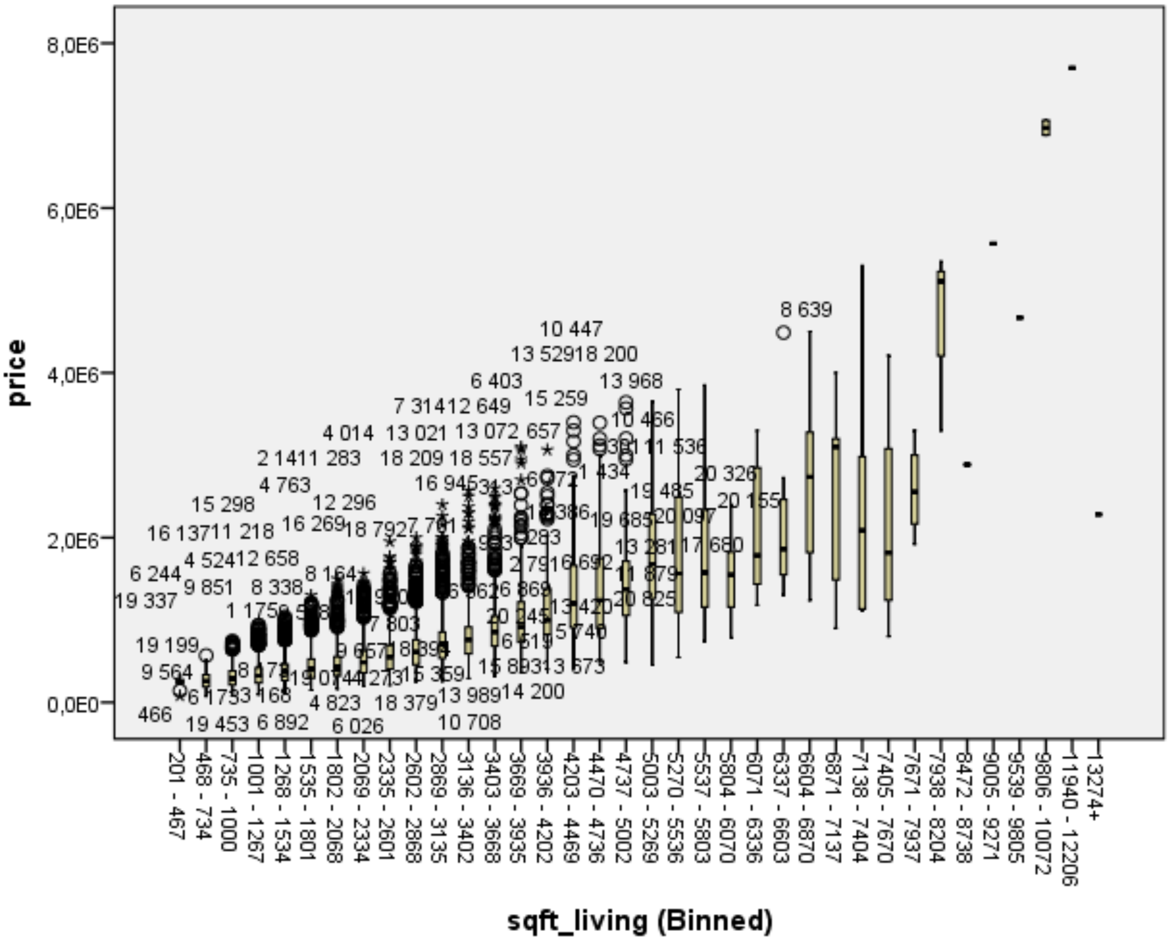


Method	Important Factors
KNN	sqft_living zipcode waterfornt floors

Factors	Method
sqft_living (*) zipcode bathrooms (*) year_built condition (*)	Decision Tree (CHAID)
sqft_living (*) waterfront condition (*) bathrooms (*) bedrooms	K-means

We thought also about another reason that led our algorithms to have those “high” errors. In fact, considering all our factors and the outlier analysis, for each of them we have lots of outliers. We take as example the box plot we did for the *Sqft_living*.

For a better visualisation of the data we organised the houses into intervals of dimension.



As we can see from the picture, also splitting our data into intervals the number of outliers is high, and we think that this “dirtiness” of our data can affect the results.

For improve the prediction we taught about two strategies:

- Clean the dataset from points that are outliers for more than 3 factors (considering we have 9 factors, it’s 1/3 of the total);
- Split the data set into two parts. In particular considering the *sqft_living,* we can split the dataset into “Little flat” and “Big flat”. We think that this kind of split can lead to a higher homogeneity of data, so maybe we can obtain better results by running all the algorithms for the 2 new groups, little and big.