Friend team project proposal: House Pricing in King County, USA

Friend team, friend@chalearn.org
Walid Belrhalmia wbelrhalmia@gmail.com, Paul-Hadrien Bourquin paulhadrien.bourquin@gmail.com, Amine Biad biad718@gmail.com
Zizhao Li sharoncarl688@gmai.com, Mo Yang lucasyeoh1992@gmail.com, Louis Trouche louisn.trouche@gmail.com.
GitHub: https://github.com/mogolola/houseprice_prediction.git

November 15, 2017

1 Background

"How much can I sell a house ?"
As a real estate agent, you will be trying to answer to this question by digging the data you have been given. In fact, this is a tricky question : you want to sell it at the higher price, but you also want to sell it and nobody will buy it if it's way overpriced. Your real estate agency in King County, a county of the state of Washington in the US, has all the records needed. You will look at the features of a lot of houses and the price they were sold at. We can guess that some properties are important, like a bigger house will probably be sold at a higher price than a smaller if they are both located in the same place, but perhaps there are some other factors not so obvious that might be taken into account when determining the price of a house. We want you to find those hidden factors that will allow you to make precise estimations of the right price of a house so we can make capital gains by buying undervalued houses and selling them thereafter.

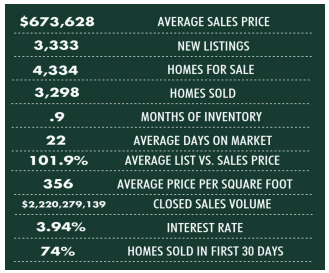

Figure 1: King country real estate market analysis

According to a study made in October 2017, the average sale price for King County homes is 673,628 dollars. In comparison 5 years ago the average sales price was 414,403 dollars. That is a 62 percent increase over 5 years ago.

2 Material and Method

The dataset used is from Kaggle, House Sales in King County, a csv file containing 21613 observations. These sales were done between May 2014 and May 2015. Each line contains the price of the house sold, and it's characteristics : number of bedrooms, bathrooms, floor surface, number of floors, waterfronts, the location and some other values. The good news is that the file looks usable from the start ! We just have to parse the csv file and retrieve the values for each column.

We chose this dataset because it is a very good example of regression model : the price depends of multiple variables between the 19 given features. Some of them might be irrelevant and have no real impact on the price of the house, while some other will have massive impacts. The goal is to make the part of what data is relevant and what is not, and from the relevant part, be able to extrapolate the correlation between the values and the price associated with them.

For the split of our data, we decided to put 60% in our training set, 20% in our validation set, and 20% in our test set.

2.0.1 Visualization and Exploration

After checking the quality of our DataSet and verify that there is no missing values, we will start by visualizing the histogram of our target variable (price):

Let's check the bivariate relationship between some variables with the housing price.

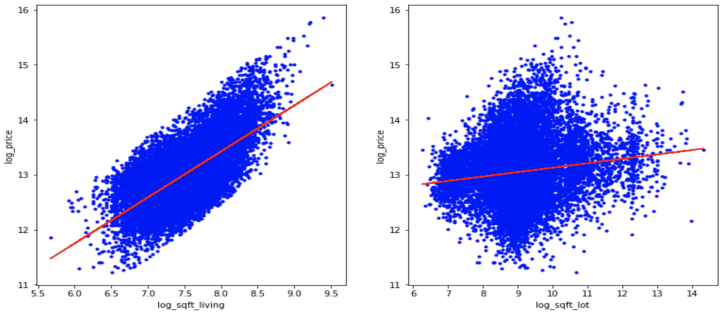

Figure 2: Bivariate relationship between area variables with the housing price

2.0.2 Finding most relevant features:

In this part, our main goal is to find variables which have a strong correlation with the house prices. To do so, we must identify two types of variables, Categorical variables and continuous variables.

Correlation for categorical variables:

By viewing the possible values of our variables , we can identify 7 categorical variables; waterfront, bedrooms, bathrooms, floors, view, condition, grade. Then, we computed the correlation of this variables with the house prices:

```
The Correlation of waterfront with price is PointbiserialrResult(correlation=0.26636943403060209, pvalue=0.0)
The Correlation of bedrooms with price is (0.308349598145633828, 0.0)
The Correlation of bathrooms with price is (0.525137505413961870, 0.0)
The Correlation of floors with price is (0.256793887550718410, 1.5810100666919889e-322)
The Correlation of view with price is (0.397293488294504280, 0.0)
The Correlation of condition with price is (0.036361789128997554, 8.935654062344094 2e-08)
The Correlation of grade with price is (0.667434256020237090, 0.0)
```

Figure 3: The correlation of categorical variables with house prices

As results shows, the top 3 categorical variables that have the highest correlation with house prices are: grade (0.66), bathrooms (0.52), view (0.39)

Correlation for continuous variables:

we will use the correlation heatmap in order to analyze the correlation of continuous variables, the top 3 continuous variables are: sqft_living (0.7), sqft_above (0.61), sqft_living15 (0.59)

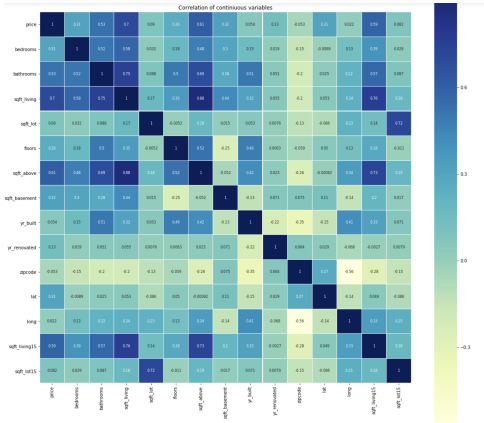

Figure 4: The correlation heatmap for continuous variables

Conclusion, from above we can determine the top 5 most correlated and important features: sqft_living=0.7, grade=0.66, sqft_above=0.61, sqft_living15=0.59, bathrooms=0.52.

Checking Data Leakage : all correlated variables with our target (price) are legitimates, and since we don't have any missing values we can conclude that we don't suffer from data leakage.

2.0.3 Advanced features ranking using Stability Selection, RFE, Random Forest, linear models:

In this part we will do some advanced features selection by combining different feature ranking methods, namely that of Recursive Feature Elimination (RFE), Stability Selection, linear models as well as Random Forest.

Stability Selection via Randomized Lasso: In a nutshell, this method serves to apply the feature selection on different parts of the data and features repeatedly until the results can be aggregated. Therefore stronger features (defined as being selected as important) will have greater scores in this method as compared to weaker features. For us, Stability Selection method is conveniently inbuilt into sklearn's randomized lasso model so we will use it.

Recursive Feature Elimination (RFE) : uses a model (eg. linear Regression or SVM) to select either the best or worst-performing feature, and then excludes this feature. The whole process is then iterated until all features in the dataset are used up (or up to a user-defined limit). Sklearn conveniently possesses a RFE function via the sklearn.feature_selection call and we will use this along with a simple linear regression model for our rank-

ing search.

Linear Model Feature Ranking: Now we will apply 3 different linear models (Linear, Lasso and Ridge Regression) and how the features are selected and prioritised via these models. To achieve this, We shall use the sklearn implementation of these models and in particular the attribute .coef to return the estimated coefficients for each feature in the linear model.

Random Forest feature ranking: Sklearn's Random Forest model also comes with it's own inbuilt feature ranking attribute and one can conveniently just call it via "featureimportances".

Ploting the features ranking: we will compute the mean ranking score attributed to each of the feature combining Recursive Feature Elimination (RFE), Stability Selection, linear models as well as Random Forest and plot that via Seaborn's factorplot:

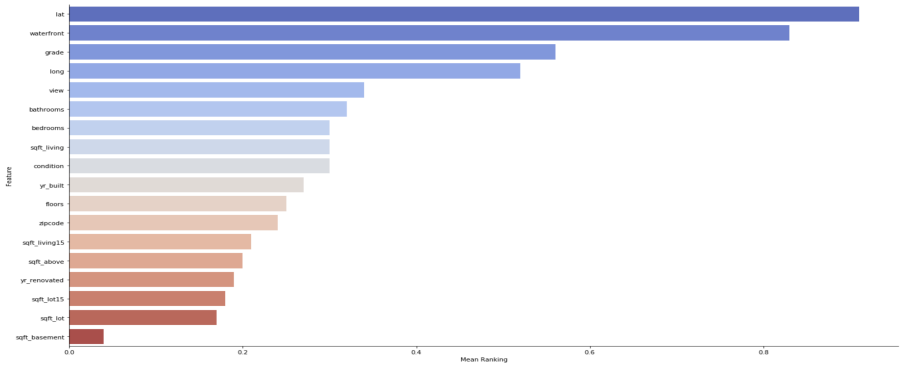

Figure 5: Features Ranking RFE, Random Forest, linear models

Well as you can see from our feature ranking, the top 3 features are 'lat', 'waterfront' and 'grade'. The bottom 3 are 'sqft_lot15', 'sqft_lot' and 'sqft_basement'. We will be using this feature selection rather than the one giving by the correlation matrix since it is more reliable and use advanced techniques.

2.0.4 Choosing our metric of success:

In order, the measure the performance of the model, we will be using the R-Squared metric, This is probably the most commonly used statistics and allows us to understand the percentage of variance in the target variable explained by the model. Its easier to interpret than other metrics because its bounded between 0 and 1(Higher is better).

3 Results:

So, here basically we will train a linear Regression Model, we will train our model on our 60% train data and test it on our validation set. So we got a result of:0.836326938583 for the R-square. Very good score, we will try to find another model that give much higher results so we compared multiples regression model and test their performances on our data.

	model	explained_variance	'neg_mean_absolute_error	neg_mean_squared_error	neg_mean_squared_log_error	neg_median_absolute_error	r2
0	decision tree	0.790989	95140.5	2.68452e+10	0.059521	54500	0.790989
1	random forest	0.868716	71167.5	1.6862e+10	0.0324396	40970.7	0.868716
2	knn	0.456893	168038	6.98201e+10	0.159745	113250	0.456395
3	MLPR	0.571144	155376	5.51004e+10	0.129732	107384	0.571
4	GradientBoost	0.884631	70619.2	1.48184e+10	0.0314796	44200.3	0.884627

Figure 6: Results on some predictive algorithms

Typically, gradient boosted decision trees(XgBoost) give the best results with an r2 error of 0.88. In order to find this value of the r2 error with the XgBoostRegressor, we needed to find the best values for the parameters. So we did a GridSearch and foud that the best values were : $colsample\_bytree : 0.7, silent : 1, learning\_rate : 0.05, nthread : 4, min\_child\_weight : 4, n\_estimators : 500, subsample : 0.7, objective : reg : linear, max_depth : 6$

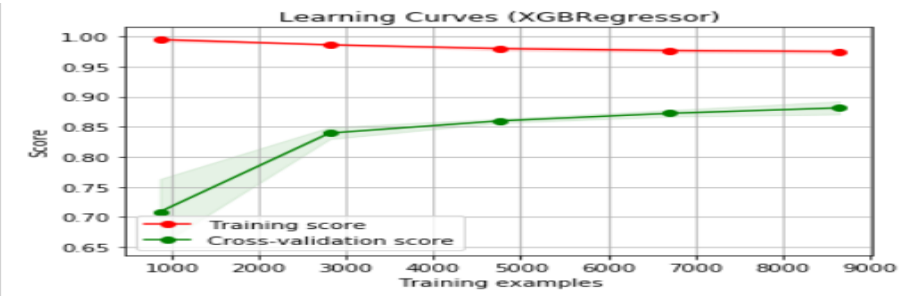

Figure 7: Learning curve of the XGBoost

We see on that figures that we obtain good results with the number of data we have in our training set so we don't need anymore data in our training set. Also We checked whether the order of our variables does not inform about the target. we done that by shuffling all our examples in random order to make sure(avoiding Data Leakage).

At the end, gradient boosted decision (XgBoost) give the best results for this dataset, and the score that we obtained is the highest among other competitors which have posted their solution about this dataset on Kaggle.