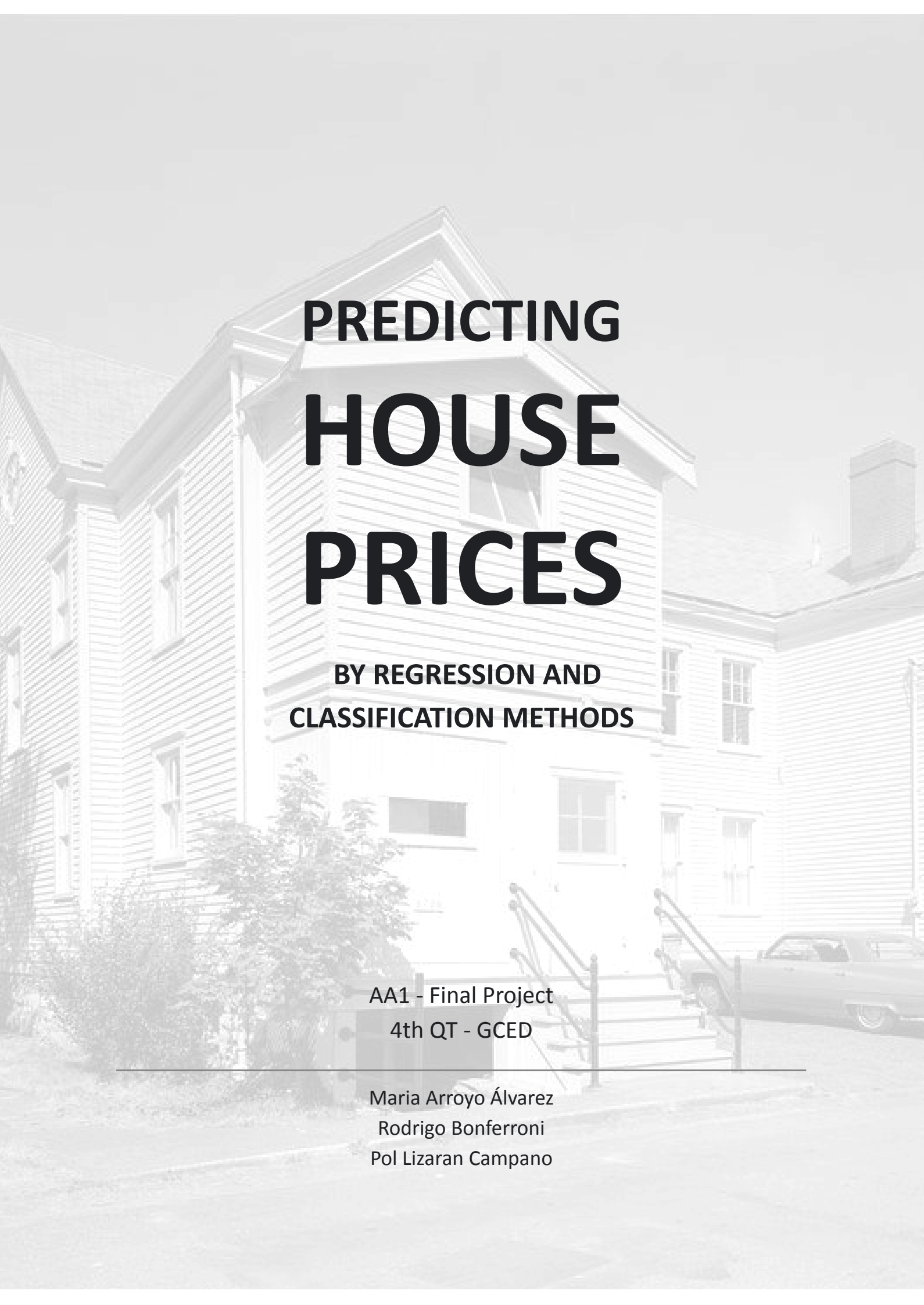


PREDICTING HOUSE PRICES

BY REGRESSION AND CLASSIFICATION METHODS

AA1 - Final Project

4th QT - GCED

Maria Arroyo Álvarez

Rodrigo Bonferroni

Pol Lizaran Campano

Index

1. Introduction

Description of the problem and motivation

Ever since this project was introduced to us, we knew we wanted to work on a topic that could tackle an interesting problem. That is why, after an exhaustive search between different data sets, we chose to work on a regression problem using the *House_sales_reduced* dataset. The main target of our study will be to predict the price of a house using information about several characteristics of it. In addition, we will also raise a classification problem: dividing into cheap and expensive houses.

This data set provides information about the features of a given set of houses which entail location, square footage, number of rooms, overall condition of the house and building details. In addition, we have available the 'price' variable, making this study a supervised one since it will be the target of our predictions.

We have chosen this topic because it seemed like an appropriate data set (fair amount of variables and enough instances) which would allow us to work without having any unexpected problems due to external factors. This will let us demonstrate the knowledge we have garnered in this course.

Information of the data set

Source: Kaggle - Harlfoxem (2016)
Link: https://www.openml.org/search?type=data&status=active&id=42635&sort=runs
Number of instances = 21613
Number of features = 21 (20 numeric & 1 categorical)

Feature description

- attribute_0: number of the instance.
- id: unique identifier of the sale and house.
- price
- bedrooms
- bathrooms
- sqft_living: square footage of the house's interior living space.
- sqft_lot: square footage of the land space.
- floors
- waterfront: whether the house is overlooking the waterfront (1) or not (0).
- view: value ranging from 0 to 4 depending on how good the view is.
- condition: value ranging from 0 to 5 depending on the condition of the house.
- grade: index from 1 to 13 depending on the building construction and design, where 1-3 falls short, 7 has an average level and 11-13 have a high quality level.
- sqft_above: square footage of the interior housing space that is above ground level.
- sqft_basement: square footage of the interior housing space that is below ground level
- yr_built: the year the house was initially built.
- yr_renovated: the year of the house's last renovation. (If none, this equals 0)
- zipcode: area where the house is located.
- lat: latitude.
- long: longitude.
- sqft_living15: the square footage of interior housing living space for the nearest 15 neighbours.
- sqft_lot15: the square footage of the land space of the nearest 15 neighbours

2. Data exploration and familiarisation with the dataset

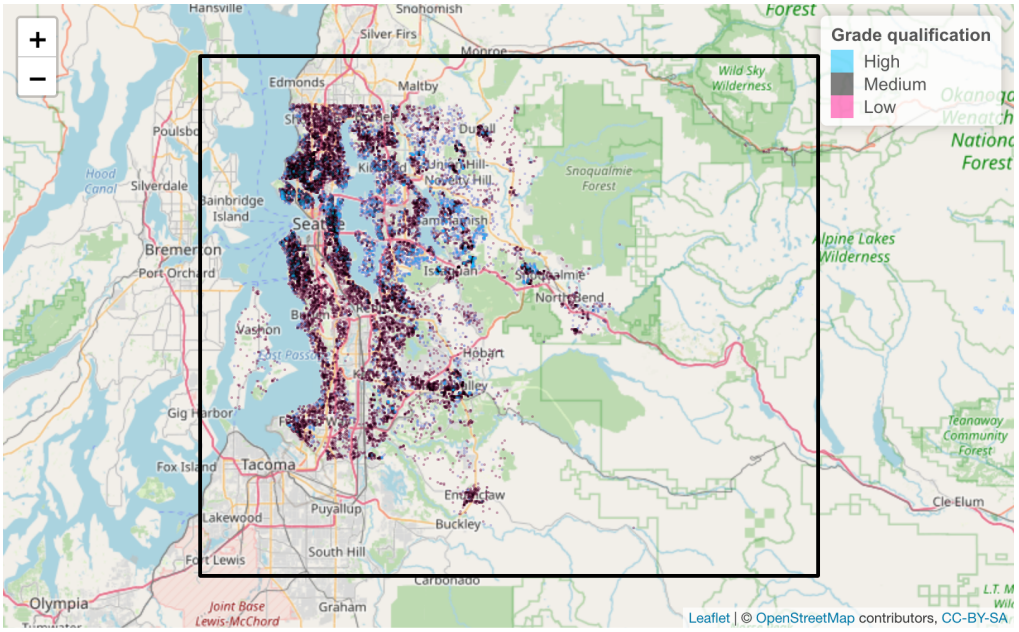
The very first step to carry out when facing a new dataset is to know which variables it contains and some qualities, such as the dimensions or the possible values for each variable and others. This way we can obtain a global view of the problem and the ways to handle it. In our case, the dataset consists of 21613 instances with 21 variables, being 20 numeric and 1 categorical (sqft_lot15), meaning that we have a mixed dataset. If we look further into 'sqft_lot15' we can notice that it has a total of 8689 distinct values. As this number could pose problems for a categorical variable, it must be transformed into a numerical variable.

Given that this dataset is from the United States, where the use of the imperial system is vastly predominant, we opt to change the units to the metric system since it will allow for easier manipulation. The names of the variables are also changed to match the previous transformation.

I. Data visualisation

In this section, our main focus is to represent the data in a significant way in order to understand it. We are going to accomplish this by plotting each sale in a map, using the available latitude and longitude variables.

HOUSES IN SEATTLE BY GRADES



As we have mentioned earlier, this dataset contains house sales produced in the USA. Concretely, in the city of Seattle (Washington), as we are able to gather from the map. To study our dataset more precisely, we have separated the data into 3 groups depending on the house's grade (high, medium and low), which we surmised from the variable with the same name.

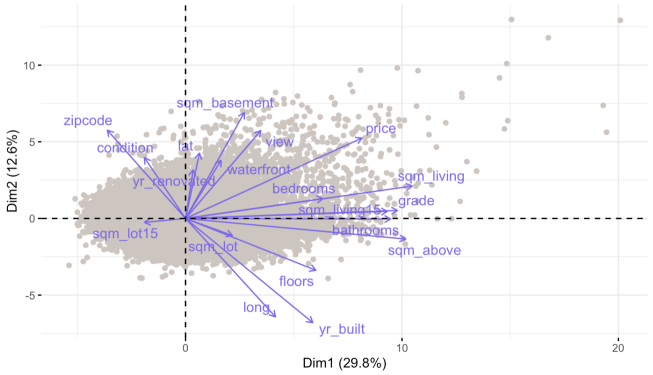
What we can deduce from taking a look at this map is that medium grade houses seem to be the most predominant. Furthermore, high grade houses are mostly concentrated in the north area whereas the other groups are mixed together.

II. Principal Components Analysis (PCA)

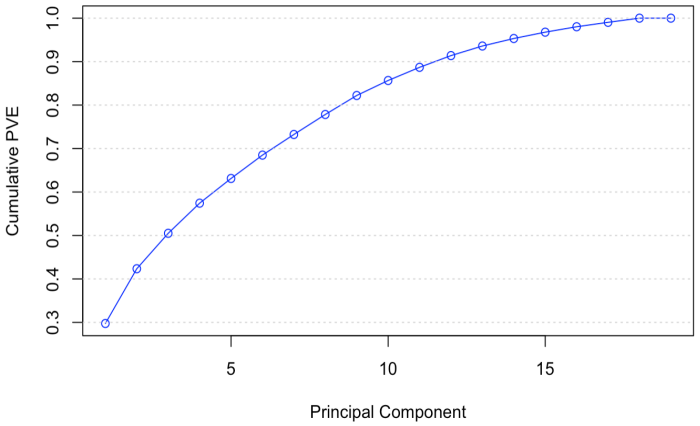
When using a dataset with multiple variables, it is important to understand how they behave all together. PCA allows us to reduce data dimensionality, so as to represent several variables in a bivariate plot. However, reducing dimensionality involves an approximation error when characterising an individual with fewer variables.

In order to apply the PCA properly we must first standardise the variables. If we fail to do so, the results won't be correct as the PCA would try to prioritise reducing the error of the variables with a higher value in comparison to the rest.

BIVARIATE PCA PLOT



PROPORTION OF VARIANCE EXPLAINED



Each arrow of the bivariate plot is called Principal Component (PC) and is an orthogonal linear combination of the original variables. In our particular case we have 19 arrows, each one representing a feature of the dataset (without taking into account 'id' and 'attribute_0'). The individuals with similar biplot scores are close in the original dimensional space. This way we are able to quickly identify some outliers, which take larger values on *price* and *sqm_living* than the others. We will study them eventually.

The proportion of the variance explained by the data with respect to the number of used principal components can be observed using the second plot. It shows that in order to explain over a 90% of the data variance at least 12 PCs are required.

III. Categorization of variables

Despite all the variables being initially defined as numerical attributes, some of them can be treated as categorical as they take one value between a limited and fixed number of possible ones. These variables are the following: 'waterfront', 'view', 'condition' and 'floors'. Along with their conversion, an assignment of a label to each of their categories is also applied.

IV. Feature extraction

Feature extraction is based on creating new variables from our current dataset to further understand tendencies. By combining the 'year_built' and 'year_renovated' variables, 'age' has been created. This new feature specifies the amount of years since the house has undergone significant changes in its structure (be it the initial construction or a renovation).

3. Resampling and data cleaning

So far, we have been modifying the entire dataset, but from now on we will solely work with a fragment of the data. With the intention of verifying how good the models proposed work, we will require a test set. That's why we have split the dataset randomly in two groups: one containing 80% of the samples (training set) and the other containing the remaining ones (test set). The training set will serve the purpose of adjusting the models, while the test set, as we've mentioned beforehand, will be used to prove how good each model is.

To predict the house prices we will be performing regression analysis, specifically, Linear or Polynomial Regression, Ridge Regression, Lasso Regression and GLM. Furthermore, we are interested in the possibility of applying a classification algorithm to this dataset and separating the houses into smaller sets. If time allows it and classification in this set is possible, we would try to implement the following algorithms: Naive Bayes, Random Forests and Decision Trees.

I. K - Nearest Neighbours (KNN)

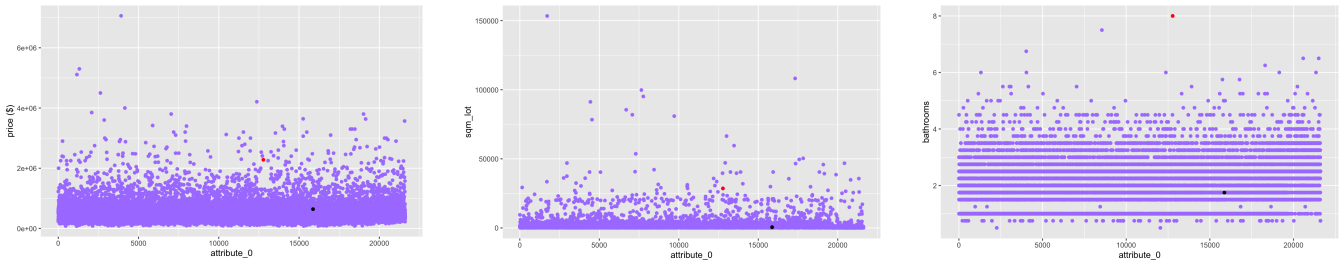
When analysing the correctness of our dataset, we observe that there are no missing values (NA). However, we realise that some values do not make sense, mainly due to unusually high or low values. This leads us to believe that some human error could have been made when creating the dataset. In particular, we notice that some houses contain zero bedrooms and/or zero bathrooms, so we'll treat them as missing values.

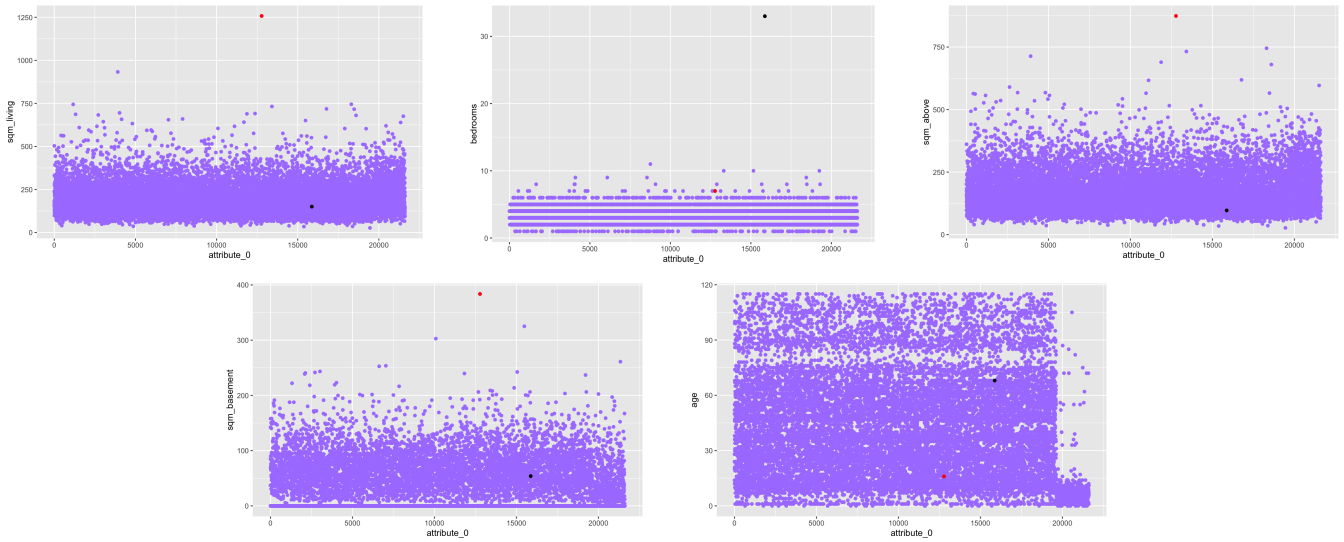
The methodology followed to correct those values has been using the k-Nearest Neighbours algorithm. This algorithm computes distance metrics among points, usually the Euclidean distance, and finds the k-nearest neighbours to the input value. Then it assigns the most common value in the k neighbours to our missing value. That is why making the right choice for the value of k is very important. Low values of k could lead to a bad classification, whereas choosing a high value could result in a large computational effort, so there is a trade-off to take into consideration.

As finding the optimal value for k might be too time-consuming, the square root of the total sample number is commonly used. Therefore, we will use $k = \sqrt{16701} \approx 129$, being 16701 the size of our training set.

II. Outliers treatment

One of the most reliable methods to find outliers is to plot different combinations of the variables in the dataset and manually check the values that you see the furthest from the pack. The following plots are the ones we have used to determine some of the outliers.



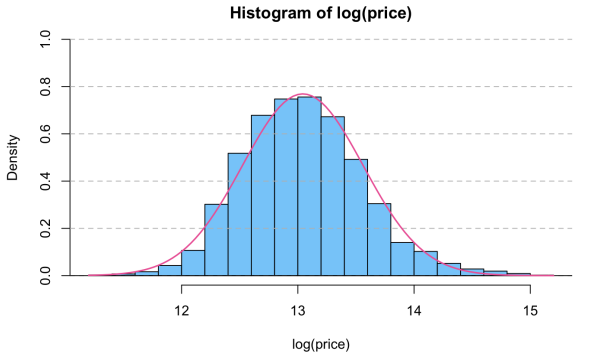


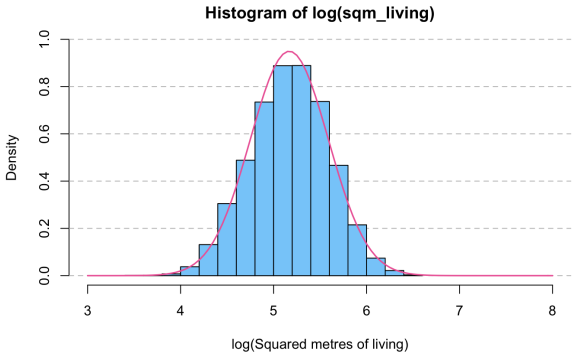
In particular, the red dot represents a house with an extremely large value of 'sqm_living', 'sqm_above' and 'sqm_basement', while the black one is a house with 33 bedrooms, which does not make any sense. For these instances, the red one has been deleted and the 'bedrooms' value for the black one has been replaced by 3, since it is considered to be a human error.

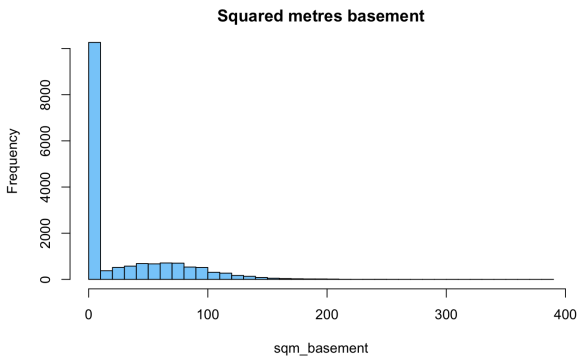
On the other hand, to identify other outliers in this dataset, we have computed the z-scores for each variable and picked the instances that have more than 3 values that are considered abnormal. A 99.8% of confidence has been used for this selection. This way 42 outliers have been found and erased from the dataset.

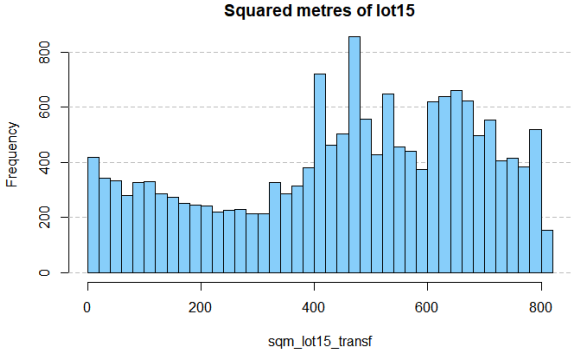
III. Gaussianization of features

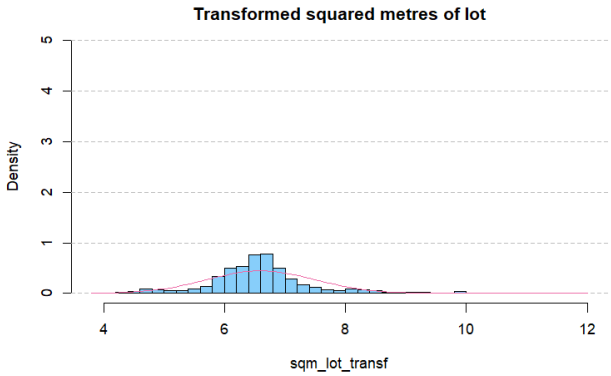
As Gaussianity is assumed in many packages that model data, we will have to check if the variables follow a Gaussian distribution and transform them if necessary. In order to do so, the Box-Cox methodology has been followed for numerical variables.











For the first two variables, we can simply apply a logarithmic transformation to obtain a normal distribution of the data. The values of 'sqm_basement' have no need for a transformation in light of the fact that they already have a normal distribution if you ignore the instances with no basement. For the variable 'sqm_above' we attempted to apply a Box-Cox transformation, but the value for lambda ended up being 1, so no transformation was actually applied and there was no improvement. Lastly, for the 'sqm_lot' variable we removed the values equal to zero and tried implementing a Box-Cox transformation, however due to the low value found for lambda (0.025) it was decided to move forward using a simple log transformation.

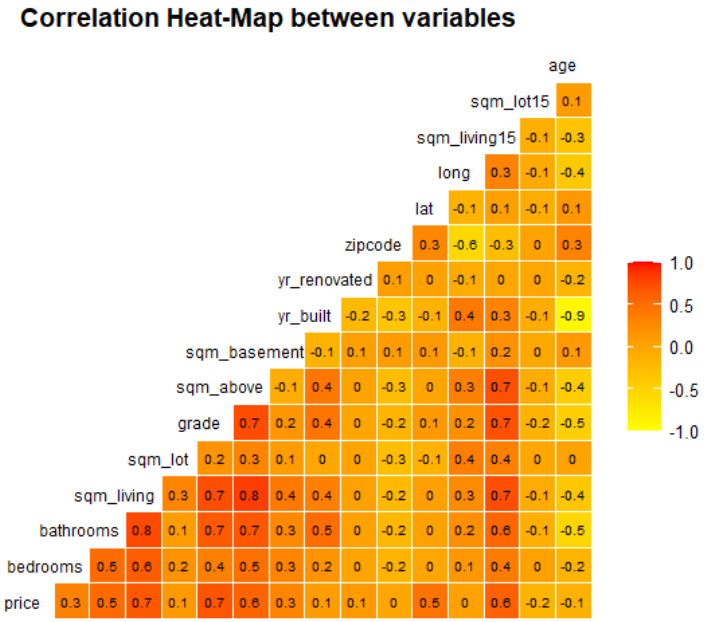
These are not the only variables that have undergone a Gaussianization transformation process, however, we have decided not to include them in this report since the Box-Cox was not working properly.

At this point, our dataset is ready to start the modelling for price prediction.

4. Modelling

I. Correlation between variables

Before starting to model our data, we have decided to make a correlation heat map on numerical features.



We observe that the most positively correlated variables are 'sqm_living', 'sqm_living15', 'sqm_above', 'grade' and 'bathrooms'. This can be due to the fact that houses with similar characteristics (such as number of floors, for example) are usually close to each other. It is the case of neighbourhoods, though sometimes really unique houses can be isolated. Moreover, it is clear that as the number of floors and square metres increase, more bathrooms are expected to be found and a better grade is normally assigned.

By contrast, we observe that the variables with higher negative correlation are 'age' and 'yr_built', being really close to -1. This is expected to happen because the more recent a house is, the lower the value for 'age' is going to be.

The main purpose of the correlation heat map is to find features with an absolute correlation value close to 1, in order to drop some when modelling. As we only have one pair of variables satisfying this, we decided not to exclude any variable when modelling since 'age' is a feature we extracted.

Previous introduction to model fitting

The aim of fitting a model is to be able to predict a numerical feature or to classify the observations into classes. Thus, some metrics are needed so as to compare the quality of the fit of the different proposed models.

As all the models we will be using are supervised ones, we are able to compute the R^2 score for the regression models and the accuracy for the classification ones. Both provide the proportion of the

target variability explained by the model, however, the R^2 is computed using the normalised root mean square error while the accuracy is directly obtained from the ratio of correctly classified among the total number of observations. The closest these metrics are to 1 the better, but as we are working with financial data, an R^2 (or accuracy) above 0.7 can be seen as a good measure of fitting quality, whereas a measure below 0.4 shows the opposite.

The data used to train all the models was the training partition mentioned earlier. Nevertheless, instead of computing the desired metrics directly, a cross validation technique was implemented. We splitted and fitted the data into 10 folds and then the mean of the different R^2 scores or accuracies was the returned value. This way, the test partition is not used until the best model is chosen.

II. Regression

We have chosen the following regression models because we have studied them in class, with the exception of the polynomial regression one. This later model was chosen after the analysis of the results of the previously chosen models, in order to find a model capable of delivering better results.

Previous to fitting the models, we can first study the Variance Inflation Factor (VIF) for each of the variables. This measure provides a measure of multicollinearity among the variables in a multiple regression model. Multicollinearity is considered problematic because the standard error of the regression coefficients gets inflated. This implies that, although the model's predictive capability is not reduced, the coefficients may not be statistically significant. We obtained the following values:

Variable	VIF
Intercept	4.8593e+06
Bedrooms	1.8444
Bathrooms	3.3226
Sqm_living	15.778
Sqm_lot	1.7814
Floors	2.4335
Waterfront	1.2042
View	1.4369
Condition	1.2627
Grade	3.3659
Sqm_above	1.3835
Sqm_basement	5.1224
Yr_built	37.589
Yr_renovated	6.8707
Zipcode	1.6697
Long	1.8721
Lat	1.2031
Sqm_living15	3.0233
Sqm_lot15	1.0753
age	37.588

The multicollinearity can be considered abnormal in variables with a VIF value higher than 10 and minimal in those with a lesser value than 5. Our variables seem to show significant multicollinearity, having up to 4 different variables with a value substantially higher than 10.

- GLM

The General Linear Model (GLM for short) is a linear regression method that consists of a continuous response variable and continuous and/or categorical predictive variables. It is the generalisation of the common linear model and includes multiple linear regression, as well as ANOVA. The benefits of using GLM over conventional regression is that it allows the dependent variable to be generated by any distribution from the exponential family and it is still quite easy to interpret.

Results of 10-fold CV with best parameters:	**GLM**
R^2 statistic	0.7691

- LASSO Regression

LASSO regression is a regularisation technique which uses the L1 norm and Shrinkage, as the word LASSO stands for Least Absolute Shrinkage and Selection Operator. L1 regularisation adds the sum of the absolute value of the magnitude of the coefficients to the residual sum of squares formula as a penalty, being λ the amount of Shrinkage:

$$\sum_{i=1}^{n}(y_i - \sum_j x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p}\left|\beta_j\right|$$

This way it prioritises sparse models with fewer parameters, being well-suited for models with high multicollinearity. It is also helpful for variable selection and parameter elimination.

Results of 10-fold CV with best parameters:	**LASSO**
R^2 statistic	0.7682

The best value for λ, which was also computed using cross validation, turned out to be 0. This means that the resulting model is equivalent to a conventional linear regression one and, therefore, LASSO is not useful with our data.

- Ridge Regression

Ridge regression is another way to implement the Ordinary Least Squares method using penalties to decrease the complexity of the model. Ridge, however, does not reduce the number of coefficients, since it never sets the coefficients' value to zero, it only minimises them. Its equation is the following:

$$\sum_{i=1}^{n}(y_i - \sum_j x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p}\beta_j^2$$

As we can see, the Ridge regression uses the L2 norm, implying that instead of using the absolute value of the magnitude of the coefficients, it uses the square power.

Results of 10-fold CV with best parameters:	**Ridge**
R^2 statistic	0.7682

The best obtained value for λ was 0.2 . However, the R^2 value is, approximately, the same as for LASSO, confirming that both models are not much better than a regular OLS linear regression.

- Polynomial Regression

A polynomial regression model is a special case of multilinear regression. This type of model is usually fit using the least squares method, in which we minimise the variance of the unbiased estimators of the coefficients. This type of regression is used when there is no linear correlation which fits all the variables. The general equation is:

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \cdots + b_n x_1^n$$

It is considered a lineal method since the regression is linear in the parameters. The number of degrees must be chosen precisely in order to properly compute the polynomial regression model. When doing so, it is important to take into account that if this value is low, the model will not be able to fit the data properly, while if it's high, the model will easily overfit the data.

Results of 10-fold CV with best parameters:	**Polynomial**
R^2 statistic	0.8250

We have chosen to use a second degree polynomial to fit the model. To do so, we have compared the results of different polynomials, up to third degree. When using a third degree we observed very poor results, which is why we decided not to continue with higher degrees.

- Gradient Boosting Regressor

The Gradient Boosting Regressor is an ensemble method based on a boosting technique, as its own name indicates. In order to predict, it uses many trees (called learners) that are ordered sequentially, so that each one can learn from the errors done by its predecessors. The amount of information learned at each step is controlled by a learning rate: the higher it is, the fewer trees will be needed to fit the model, but also the higher probability of overfitting.

Results of 10-fold CV with best parameters:	**Gradient Boosting Regressor**
R^2 statistic	0.8976

Instead of using the *GradientBoostingRegressor* function, we used the updated and more efficient *HistGradientBoostingRegressor* one, which performs binning on continuous features.

To choose the best of parameters for this model we compared the R^2 value of each combination. As we are using cross validation, we are avoiding the possible overfitting and therefore we do not need to be wary of choosing the model with the highest R^2. That model has the following parameters:

- `learning_rate` (amount of learned information) = 0.1
- `max_depth` (maximum number of levels allowed by tree) = 10
- `max_bins` (maximum of bins when binning continuous features) = 255

III. Classification

The reasons for selecting the next models are varied. Most models, such as Logistic regression, SVM, Decision Trees, Random Forest and Gradient Boosting, were mentioned during the theoretical portion of the course. The LDA and QDA methods have also been studied, although in a different subject (Data Analysis). Lastly, the Naive Bayes model was found when researching for models with a similar functioning to the LDA and QDA ones.

- *Linear classification*
    - Logistic Regression

Logistic regression is a binary classification type of GLM that, instead of using the Gaussian distribution and the identity as a link function (which was the regression case), it considers the Bernoulli distribution and uses the logit function.

Results of 10-fold CV with best parameters:	**Logistic Regression**
Accuracy	0.5650

This accuracy could be due to the fact that 'price' has no linear correlation with the other features. In such cases, Logistic Regression does not predict with a good accuracy.

- LDA (Linear Discriminant Analysis)

This is a probability model that works under Bayes theorem. Given a set of characteristics, it computes the posteriori probability of an observation belonging to the k-th group. What is usually done with these methods is to assign the group that maximises the probability a posteriori, that is, the one that minimises the probability of classification error. To do so, it is assumed that the covariance matrix (Σ_k) is the same for all groups.

Results of 10-fold CV with best parameters:	**LDA**
Accuracy	0.8371

- Naive Bayes

The model proposed just above, comes from the Naive Bayes model. Unlike LDA, it assumes that the effect of a particular feature on a class is independent of others and hence its name Naive. Due to its definition, it works quite fast and, under independence conditions, it tends to report better results than Logistic Regression. Furthermore, it has been seen that, when comparing numeric variables with categorical ones, it performs good results.

Results of 10-fold CV with best parameters:	**Naive Bayes**
Accuracy	0.7570

- SVM (Support Vector Machine)

A support vector machine is a linear method that is widely used for binary classification. Its objective is to fit a hyperplane separating both classes. Once it is fitted, two extra parallel planes (one at each side) are defined at the same distance, trying to maximise the margin of separation between them.

In case of having non linearly separable data, the SVM allows a relaxation on the restrictions by adding slacks to the model. They are defined as the error distance with regard to the hyperplane, so that the total squared sum of them is minimised too.

Results of 10-fold CV with best parameters:	**SVM**
Accuracy	0.8409

It is important to add that, although we obtained a good enough accuracy, this method had convergence problems when using the default tolerance value. This value had to be reduced in order for it to work properly.

- *Non-linear Classification*

  - QDA (Quadratic Discriminant Analysis)

As seen beforehand, Discriminant Analysis makes use of Bayes theorem. Thus, QDA works exactly like LDA with the only difference being that it assumes that the covariance matrix ($\sum_k$) is different for all groups, making it a non-linear model.

Results of 10-fold CV with best parameters:	**QDA**
Accuracy	0.7446

  - Decision Tree Classifier

Decision trees are a data structure formed by a root (first node), branches (decision rules) and leafs (outcome of the model). Once the decision tree has been created, this model asks binary questions (Yes/No) based on the characteristics of an observation. Depending on the answer, it follows one branch or another until a leaf node is reached, which will determine the assigned class.

Results of 10-fold CV with best parameters:	**Decision Tree Classifier**
Accuracy	0.8667

  - Random Forest

Random Forest is an ensemble method consisting of many uncorrelated Decision Trees working with different partitions of the data, thus being a bagging model. Its prediction is based on the idea of *wisdom of crowds,* which assigns the most “voted” answer using the different Decision Trees. The main advantage is that it controls the possible overfitting that can be caused by a single Decision Tree when using multiple features.

Results of 10-fold CV with best parameters:	**Random Forest**
Accuracy	0.9088

Best set of parameters:

- `max_depth` (maximum number of levels allowed by tree) = 10
- `max_features` (maximum of features taken for splitting each node) = 0.6
- `max_samples` (maximum of samples for defining a tree) = 0.55
- `n_estimators` (number of trees in the forest) = 80

- Gradient Boosting Classifier

Both Gradient Boosting Regressor and Classifier work following the same methodology. However, as the latter is used to predict classes, the loss function of the learners is the log-likelihood instead of the Mean Squared error.

Results of 10-fold CV with best parameters:	**Gradient Boosting Classifier**
Accuracy	0.9114

Best set of parameters:

- `max_depth` (maximum number of levels allowed by tree) = 7
- `max_features` (maximum of features taken for splitting each node) = 0.4
- `learning_rate` (amount of learned information) = 0.1
- `n_estimators` (number of trees in the forest) = 80

IV. Model comparison

After gathering all the results, it is time to compare them and choose the models that best fit our data, in order to validate them with the testing partition of our dataset.

For the regression analysis, it is clear that the Gradient Boosting Regressor model is the one with a better R^2 value. This may be due to the fact that it is a non-linear model. However, we must take into account that the Gradient Boosting method easily overfits in datasets with "noise" (measurement and sampling errors) so this might not be the ultimately best one when fitting the data.

On the other hand, we decided to choose both a linear and a non-linear model for the classification task. Among the linear ones, LDA and SVM have really similar values of accuracy, but we choose to work with LDA as we did not encounter convergence issues when using it. For the non-linear ones, Random Forest and Gradient Boosting Classifier displayed the best results. The latter tends to overfit data whenever a bad learning rate is selected, while Random Forests are more likely to smooth its effect due to the fact that they classify according to the *wisdom of crowds* criteria. Since these two methods display really similar accuracies and we can not consider that Gradient Boosting is not overfitting, we will test both of them just to be sure.

V. Test validation

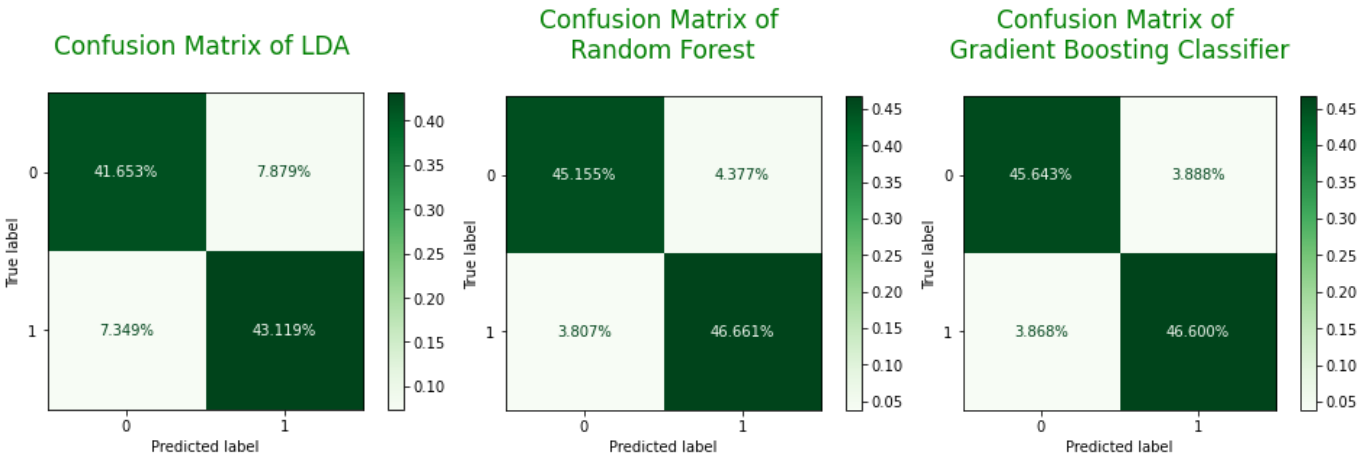
Once we have chosen our final models, we will use the test data partition to validate them. This step is crucial in order to be able to make robust predictions, otherwise, the credibility of our models would be non-existing. It is important to mention that test validation is considered to be an out-of-sample validation since a completely new dataset (not used when fitting) is used to study the

goodness of fit. In-sample validation, on the contrary, was the methodology followed during model fitting when using cross validation, as a fraction of the training dataset is used to validate.
The obtained testing scores were the following:

Test Score	Gradient Boosting Regressor	LDA	Random Forest	Gradient Boosting Classifier
Value	0.9	0.8477	0.9182	0.9224

Bear in mind that when considering the Gradient Boosting Regressor score we use the R^2 statistic instead of accuracy, which is what we apply when considering a classifier. The classification method with the highest score is the one based on Gradient Boosting, which leads us to believe that this might be in fact the best way to classify our data. Using these newly acquired test scores we can also confirm that neither of the ensemble methods seems to be overfitting the data.

Additionally, we also generate the corresponding confusion matrices for the classification methods:



A confusion matrix allows us to visualise the performance of the algorithms in a way that we can observe what type of error is the most commonly made. While Gradient Boosting and Random Forest are very similar in terms of overall accuracy, it seems like the Random Forest is more likely to commit a Type II error (classifying a cheaper house as an expensive one). As a matter of fact, Gradient Boosting is the one out of the three which has the most similar error value for both types.

In conclusion, our final model selection will consist of both Gradient Boosting methods, since they have been proven to be the best among their section. The LDA method will also be included, because we consider that it being a linear and non-ensemble method will add variety and prove to be useful as well.

5. Personal prediction

With the purpose of studying situations in which home sales companies could make use of these models, we have carried out experiments to predict how our houses would be valued or classified in the hypothetical scenario that they were located in Seattle. That is to say, we have created a new observation for the dataset containing all the information corresponding to the characteristics of a group member's home and predicted its expected price and group. Arbitrary values based on the range of our dataset for the zipcode, lat and long variables were assumed in order to "position" our house in Seattle.

For these predictions, the models that are planned to be used are the same ones as in the testing phase with the exception of Random Forest, as Gradient Boosting was considered better for the non-linear classification task. The results obtained from them were the following ones:

- ***Regression***

Regression predictions	Gradient Boosting Regressor
High-priced area	1623806 $ —> 1543858 €
Low-priced area	843650 $ —> 802113 €

Taking 1 $ ≃ 0,95 €

It is expected that a house located in a high-priced area will be more expensive than a similar one in a low-priced area, as we have seen before that the variables related to the location (*zipcode*, *long*, *lat*) are all significant when defining models.

However, these prices should be carefully considered, as the house we used as a reference is actually located in a town and not a big city as Seattle is. This implies that a square metre is more expensive and maybe it is not that common to have a garden.

- ***Classification***

Classification predictions	LDA	Gradient Boosting Classifier
High-priced area	Cheap	Expensive
Low-priced area	Cheap	Expensive

Seeing that the two models give contradictory results, we proceed to also make a prediction with the Random Forest model to assert which is the correct classification:

Classification predictions	Random Forest
High-priced area	Expensive
Low-priced area	Expensive

We can now conclude that our house would be considered expensive, regardless of the area it is located. This makes sense, as we know that the price median for the training dataset (which was the established boundary between the two classes) was about 450000 $ and both the predicted prices by the regression model are higher than that.

6. Conclusions

We have created models capable of predicting the value of a house with great precision. However, we believe that some problems stemmed from our dataset. Concretely, we think that although the price of a house does depend on the variables we had available, there is a large fraction of the value that cannot be easily explained using solely these variables. That is without mentioning the fact that house prices are prone to fluctuations. Another reason why the dataset was not ideal for an analysis is its unreliability, as many instances were found to have incorrect values. Finding and correcting said values is not realistically possible considering it has over 21 thousand observations.

Furthermore, to execute our classification portion of the project we had to create an artificial partition of the 'price' variable. To perform an ideal classification process, it would have been preferable to have a dataset with a categorical variable as the target to predict. Even so, the obtained results were pretty good.

Aside from the problems arising from the dataset, we also encountered a fair amount of difficulty when fitting models to our data. The VIF analysis of our variables showed unusual values of multicollinearity, which was expected seeing that many variables were highly correlated between them. This seemed to indicate a great functioning of models like GLM, LASSO and Ridge which, as we later observed, was not the case. The LASSO and the Logistic Regression model also had convergence problems, which we were unable to solve even when lowering the tolerance for the stopping criteria and raising the maximum number of iterations.

Moreover, many parameters had to be chosen for the creation of classification models. These parameters had to be arbitrarily chosen, so a previous study was needed in order to define them properly, as well as an implementation of a cross-validation technique to select them among a range of candidates.

What we have come to realise with our study is the importance of a proper preprocessing of the data, since it directly affects the final results. We were also able to discover different methods aside from the ones seen in class, thus further expanding our knowledge. This also showed us the vast superiority of Ensemble methods like Random Forest and Gradient Boosting. These methods were capable of easily surpassing the score of regular ones, though they took more time to perform.

This project could be further worked on to ensure its usability. One aspect that could be improved is the possibility of adding new variables, corresponding to the current economic affairs (state of recession, economic crisis, etc.) which can largely affect the final price of a house. Another aspect that could be added is modelling a neural network. This could not be done, since it was not studied during the course and that made it highly difficult to implement.

Despite all of the problems encountered, we are really satisfied with the final results of our study and we have the feeling that we have learned plenty during the process of making it.

7. References

- Artful Analytics.(2020, July 8). *A Feature Preprocessing Workflow*. R bloggers. https://www.r-bloggers.com/2020/07/a-feature-preprocessing-workflow/
- Amey Band.(2020, May 23). *How to find the optimal value of K in KNN*. Towards Data Science. https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb
- SaiGayatri Vidali.(2017, Dec 29). *Day 8. Data Transformation - Skewness, normalisation and much more*. Medium. https://medium.com/@TheDataGyan/day-8-data-transformation-skewness-normalization-and-much-more-4c144d370e55
- Pavan Vadapalli.(2020, Jul 27). *6 Types of Regression Models in Machine Learning You Should Know About*. upGrad. https://www.upgrad.com/blog/types-of-regression-models-in-machine-learning/#1_Linear_Regression
- iSixSigma.(2020, Apr 1). *Overview: What is GLM?*.iSixSigma. https://www.isixsigma.com/dictionary/general-linear-model-glm/
- Dinesh Kumar.(2021, Dec 26). *A Complete understanding of LASSO Regression*. GreatLearning. https://www.mygreatlearning.com/blog/understanding-of-lasso-regression/
- Abhishek Sharma.(2020, Mar 16). *Introduction to Polynomial Regression (with Python Implementation)*. AnalyticsVidhya. https://www.analyticsvidhya.com/blog/2020/03/polynomial-regression-python/
- Scikit-Learn developers.(2007-2022). *Linear and Quadratic Discriminant Analysis*. Scikit-Learn. https://scikit-learn.org/stable/modules/lda_qda.html
- George Lawton. (2022, Jan). *Logistic Regression.* TechTarget. https://www.techtarget.com/searchbusinessanalytics/definition/logistic-regression
- Alon Lekhtman. (2021, Feb 19). *When logistic regression simply doesn't work.* Towards Data Science. https://towardsdatascience.com/when-logistic-regression-simply-doesnt-work-8cd8f2f9d997
- Pol Ligdi Gonzalez.(2019, Sep 20). *Naive Bayes - Teoría*. aprendeIA. https://aprendeia.com/naive-bayes-teoria-machine-learning/
- JavaTPoint.(2011-2021). *Decision Tree Classification algorithm*. javaTpoint. https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
- Tony Yiu.(2019, Jun 12). *Understanding Random Forests*. Towards Data Science. https://towardsdatascience.com/understanding-random-forest-58381e0602d2
- Anshul Saini.(2021, Sep 20). *Gradient Boosting Algorithm: A Complete Guide for Beginners*. AnalyticsVidhya. https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/
- Stephanie Glen.(2019, Jul 28). *Decision Tree vs Random Forest vs Gradient Boosting Machines: Explained Simply*. TechTarget. https://www.datasciencecentral.com/decision-tree-vs-random-forest-vs-boosted-trees-explained/
- pankaj.0323.(2022, Jan 15). *Python predict() function - All you need to know!*. AskPython. https://www.askpython.com/python/examples/python-predict-function
- David Cartyi.(2021, Apr 29). *Training Data vs. Validation Data vs. Test Data for ML Algorithms*. APPLAUSE. https://www.applause.com/blog/training-data-validation-data-vs-test-data