Statistical Methods - Final Project: House Prices - Regressing Sales Price

Donninelli Adriano Insaghi Edoardo Zappi Piero Zappia Edoardo

Introduction

This project aims to develop statistical models capable of predicting house sale prices from a set of covariates describing every aspect of the property. To do it we leverage the "House Prices - Advanced Regression Techniques" dataset from Kaggle [1]. This dataset comprises 1460 samples of property sales situated in Ames, Iowa. For each one, 79 explanatory variables are collected, varying from the year of construction to the type of foundations used.

We begin by exploring the data and doing the necessary preprocessing steps. After that, we simplify the data using a dimensionality reduction technique (PCA) to gain insights into how the target variable relates to other factors. Next, we suggest up to four statistical models: LM, GAM, RF, and a basic NN. We performed some tests with Generalized Linear Models using the Gamma distribution as well but did not obtain good results. We fine-tune each model and compare their behaviours using cross-validation since there is no predefined validation dataset at disposal.

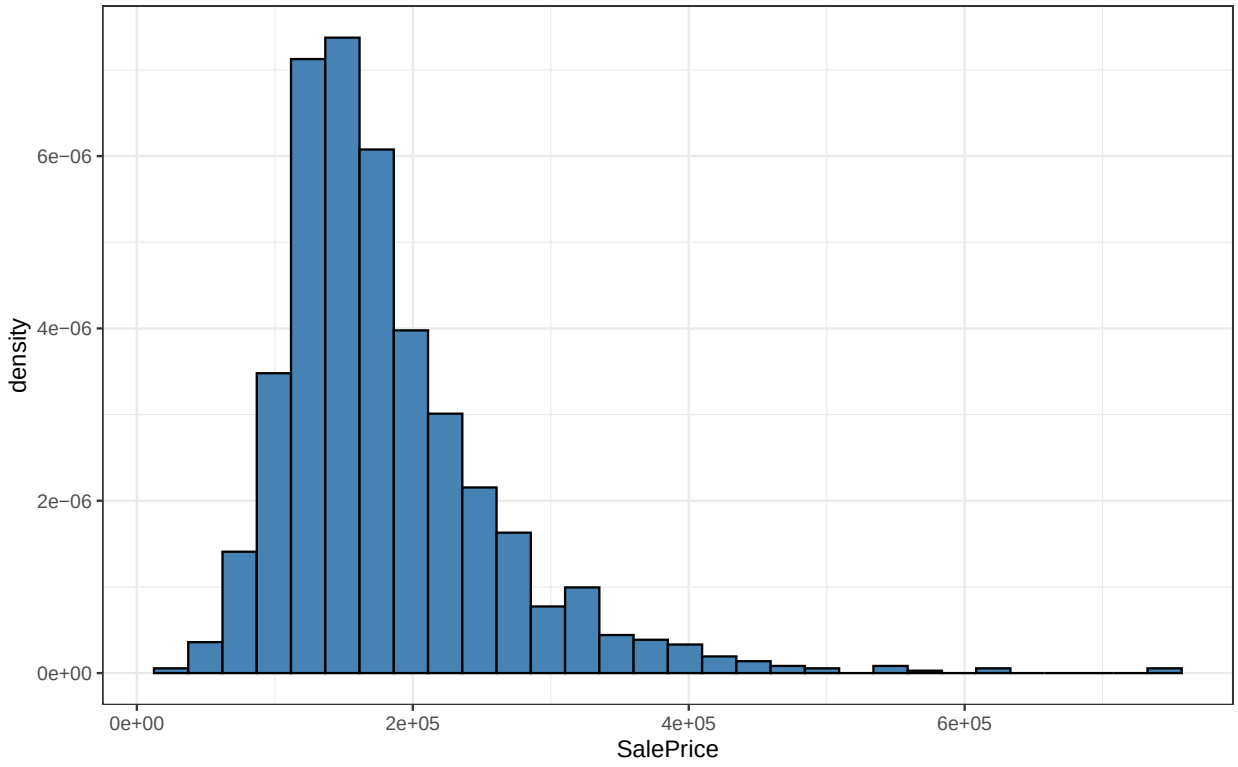
Data Exploration

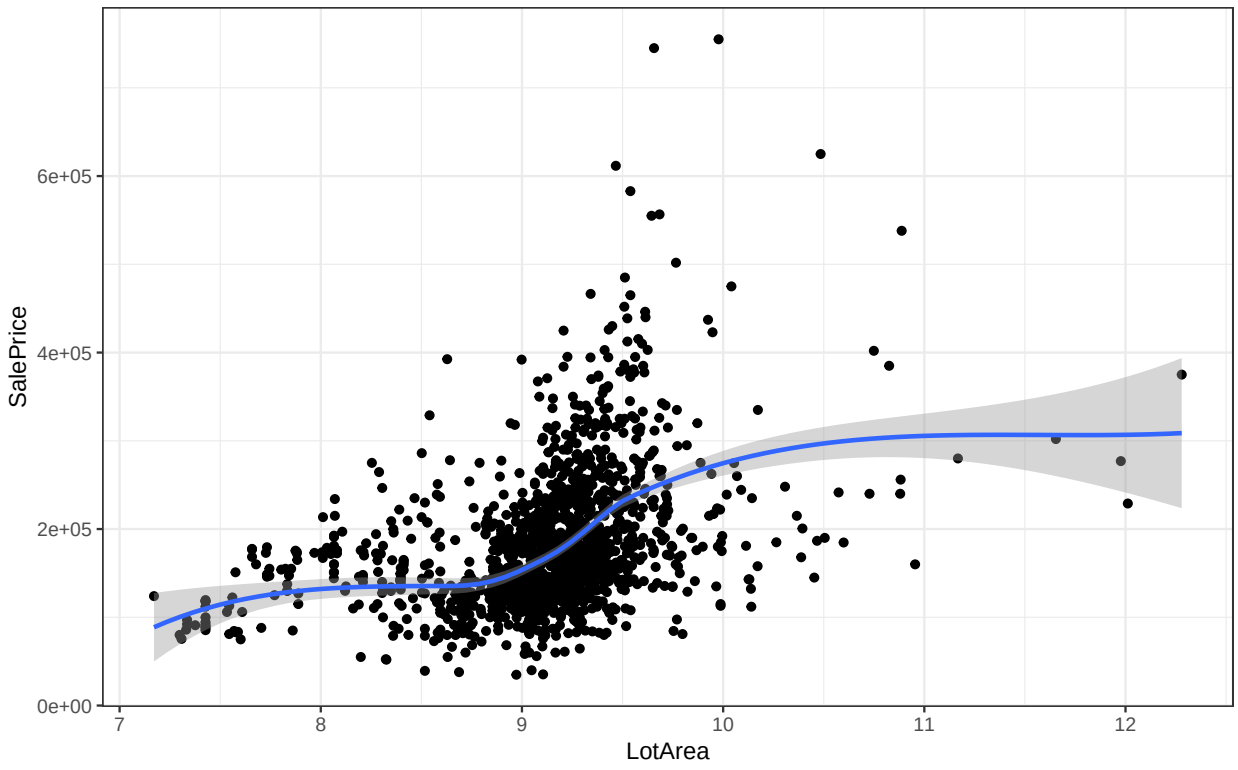
The first step is to explore the data. As mentioned in the introduction, the dataset consists of 1460 observations and 79 covariates, which is a substantial number of variables, and poses the challenge of selecting which ones play a relevant role in determining the response variable.

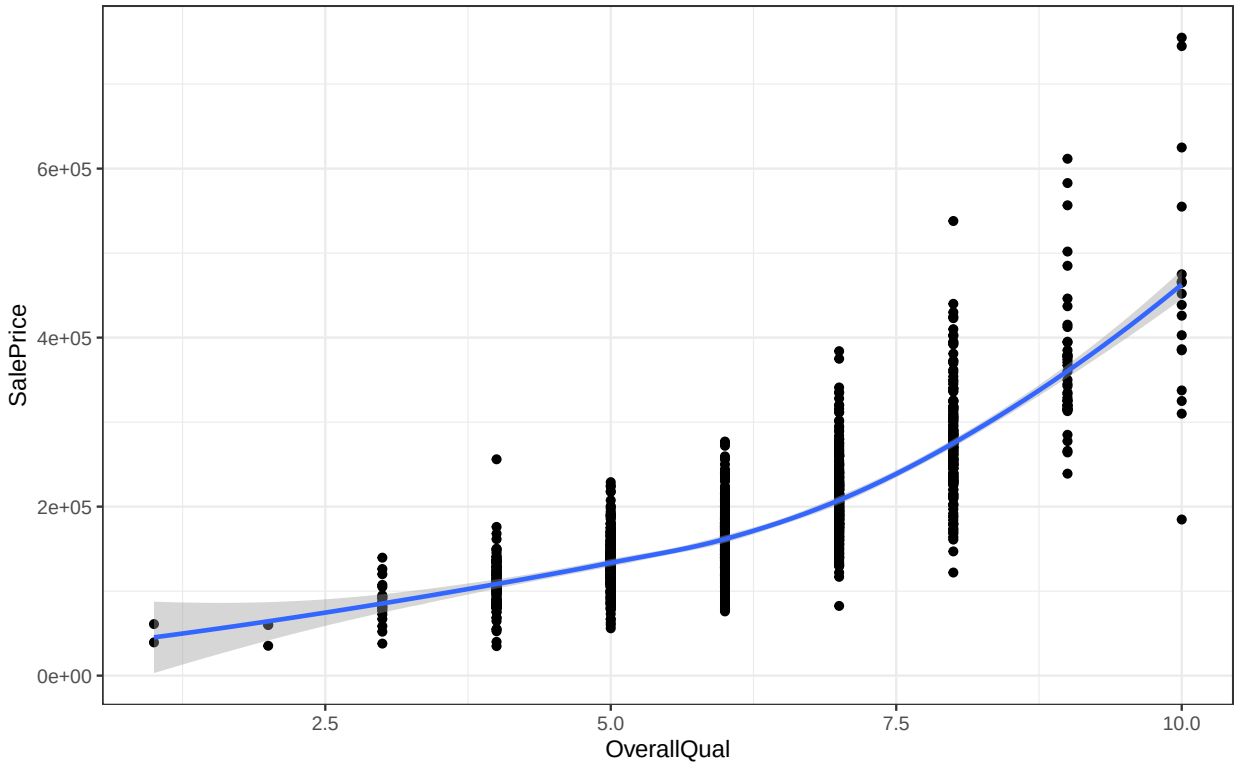
The preprocessing of the data consisted in a number of steps: we first decided to remove from the dataset all the variables with an excessive number of NAs and fill those which had a reasonable amount of NAs (<100) with the mean or the mode of the remaining observations, depending on whether such variables were numerical or categorical. We then proceeded to scale the extremely right skewed numerical variables using a logarithmic transformation, which helps normalizing the variables increasing the explainability of the models. Lastly, given the substantial number of categorical variables with dozens, if not hundreds, of categories, we decided to drop the least occurring ones into a separate category. This procedure helps keeping the degrees of freedom under control and makes the models easier to explain.

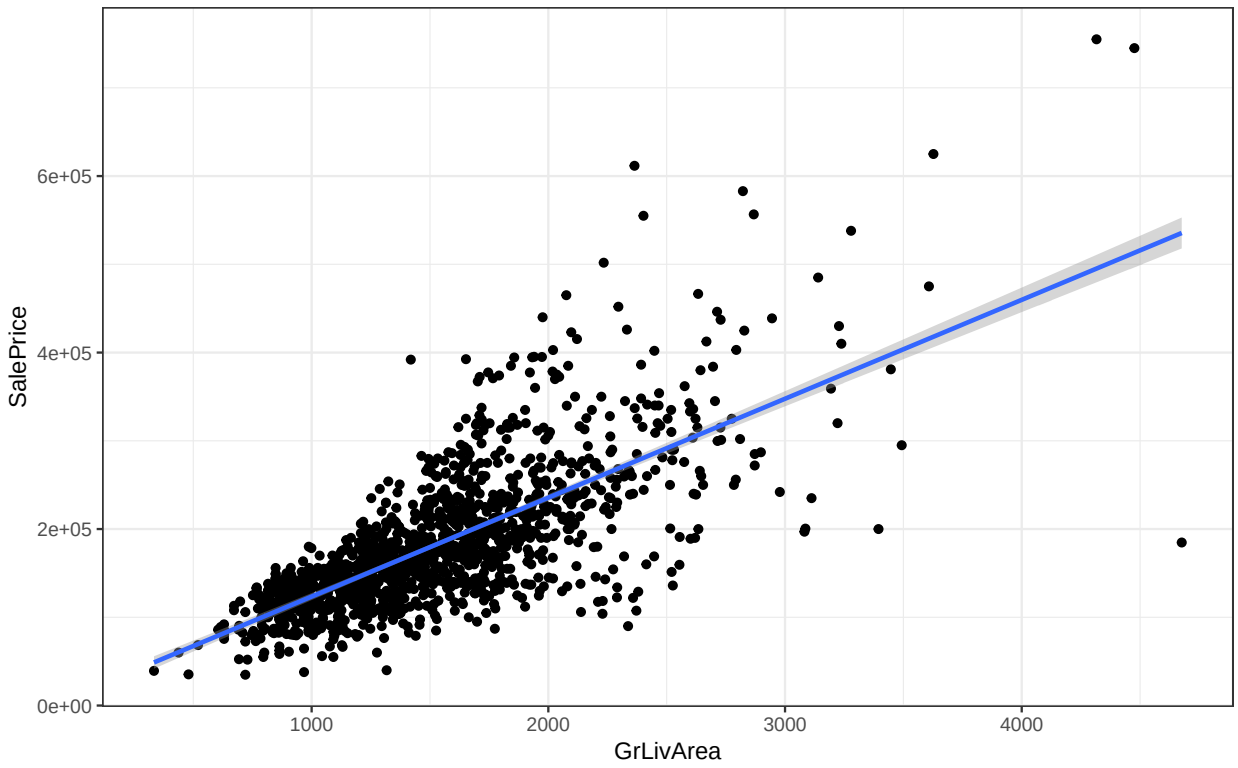
We considered taking the logarithm of the response variable, which appears to be right skewed, but this does not improve significantly the explanatory power of the models. We therefore decided to not change it, preserving the interpretability of the models while adhering to Occam's Razor principle.

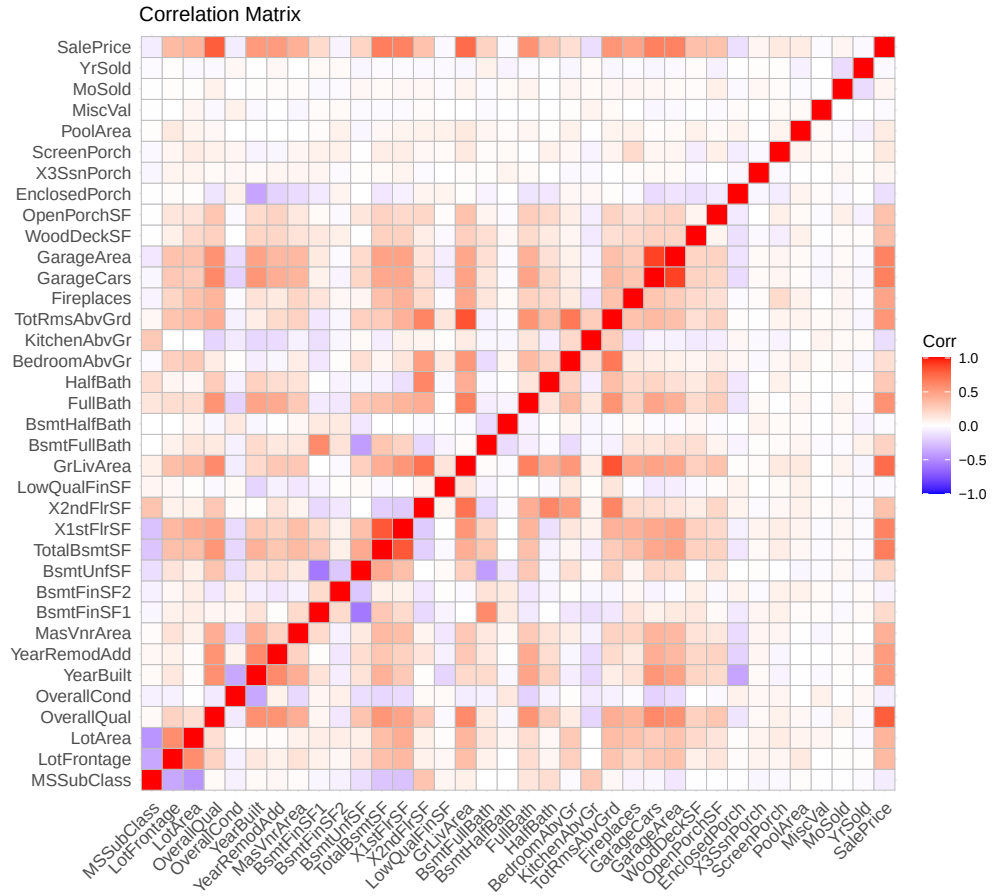
We observed the correlation matrix to explore how the numerical variables are related to each other and have a first look at how they are correlated to the response variable. We then show a few scatterplots of the response variable against some interesting covariates, looking for the relationship with the house price.


density
6e−06
4e−06
2e−06
0e+00
0e+00
2e+05
4e+05
6e+05
SalePrice


SalePrice
6e+05
4e+05
2e+05
0e+00
7
8
9
10
11
12
LotArea


SalePrice
6e+05
4e+05
2e+05
0e+00
2.5
5.0
7.5
10.0
OverallQual


SalePrice
6e+05
4e+05
2e+05
0e+00
1000
2000
3000
4000
GrLivArea



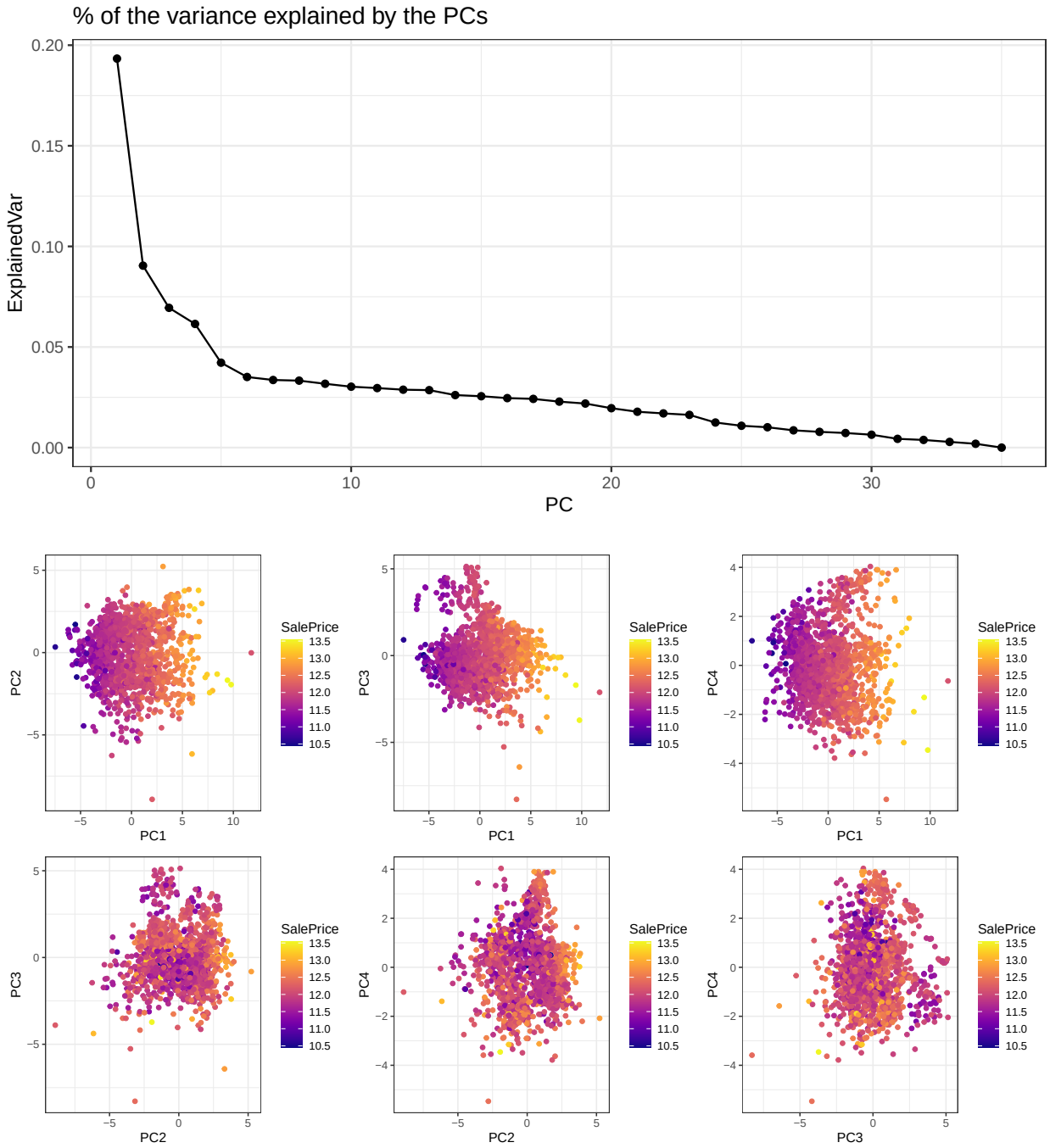
Dimensionality Reduction

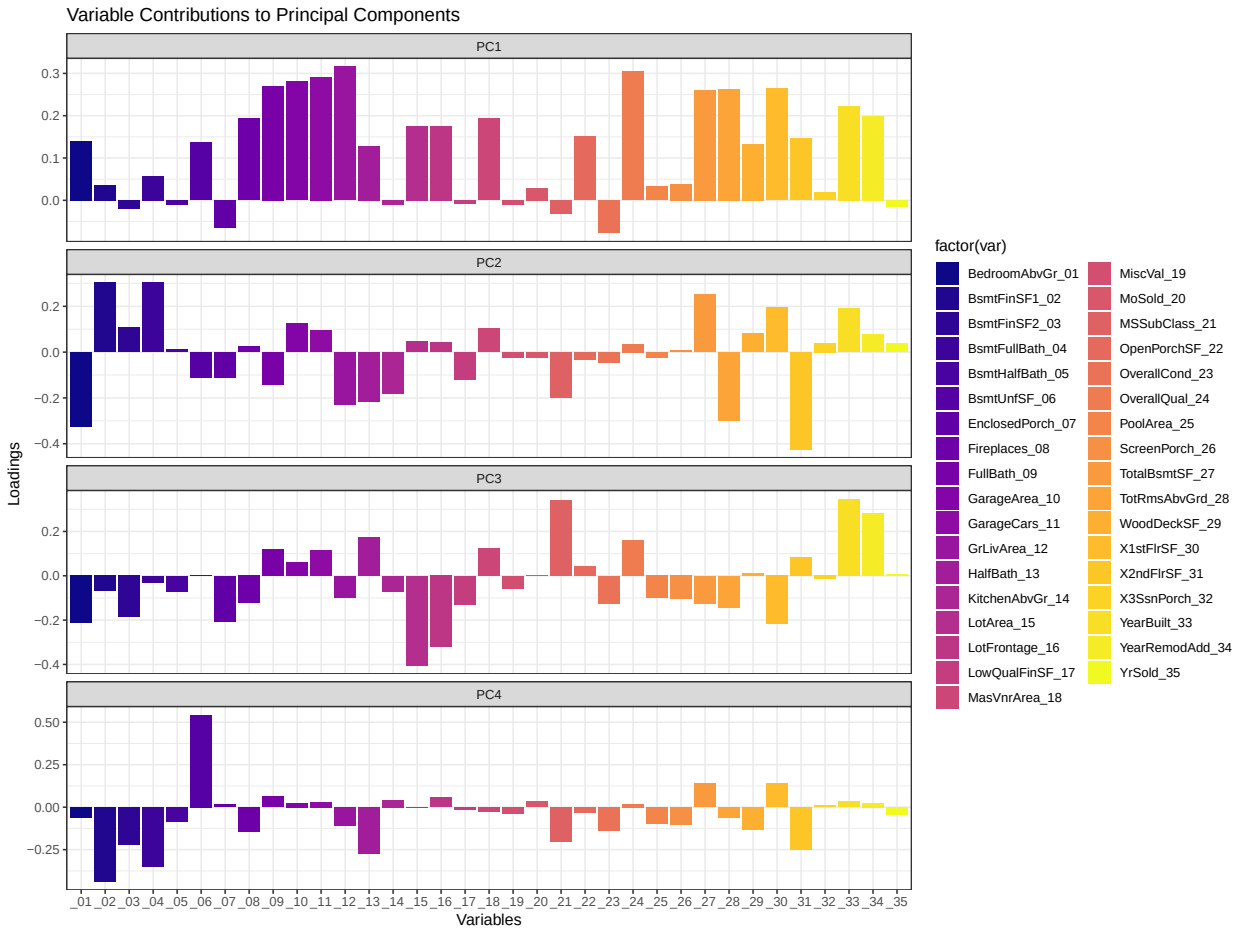
When the number of covariates is high, it may be beneficial to put in place techniques that aim at reducing the dimensionality of the dataset, while preserving as much information as possible. We decided to use principal component analysis, a linear dimensionality reduction tool that works by projecting the data onto the eigenvectors of its covariance matrix, in order to see whether the dataset can be summarized with less variables.

The elbowplot in the figure represents the proportion of the variance explained by each principal component. The results are not extremely exciting, because ideally we would like to see the first few principal components explaining the majority of the variance in our data, which is not what is happening here.

We then plotted the first four principal components against each other, highlighting the value of the Sale Price in the plots, looking for interesting patterns. Apparently, only the first principal component seems to explain something about the response variable.

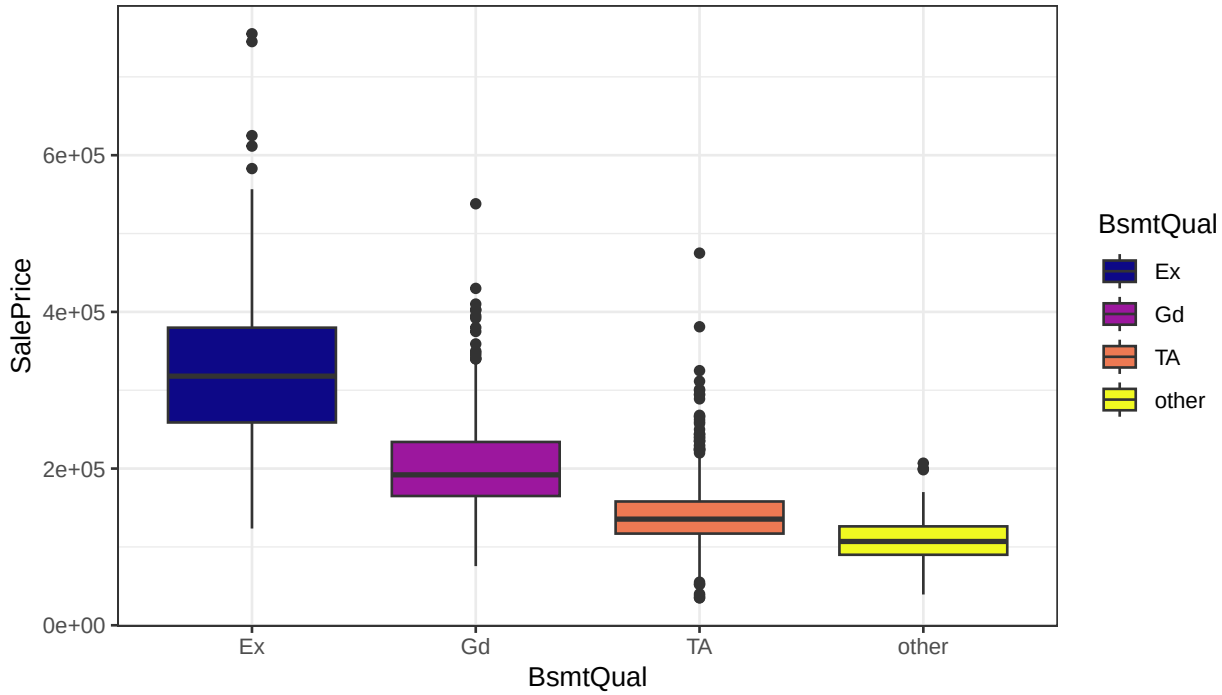
The last plot includes the loadings of the variables on the first few principal components, here we focus our attention on which variables contribute noticeably to the first component. These variables have to be observed closely since they are likely to influence the price of the house.

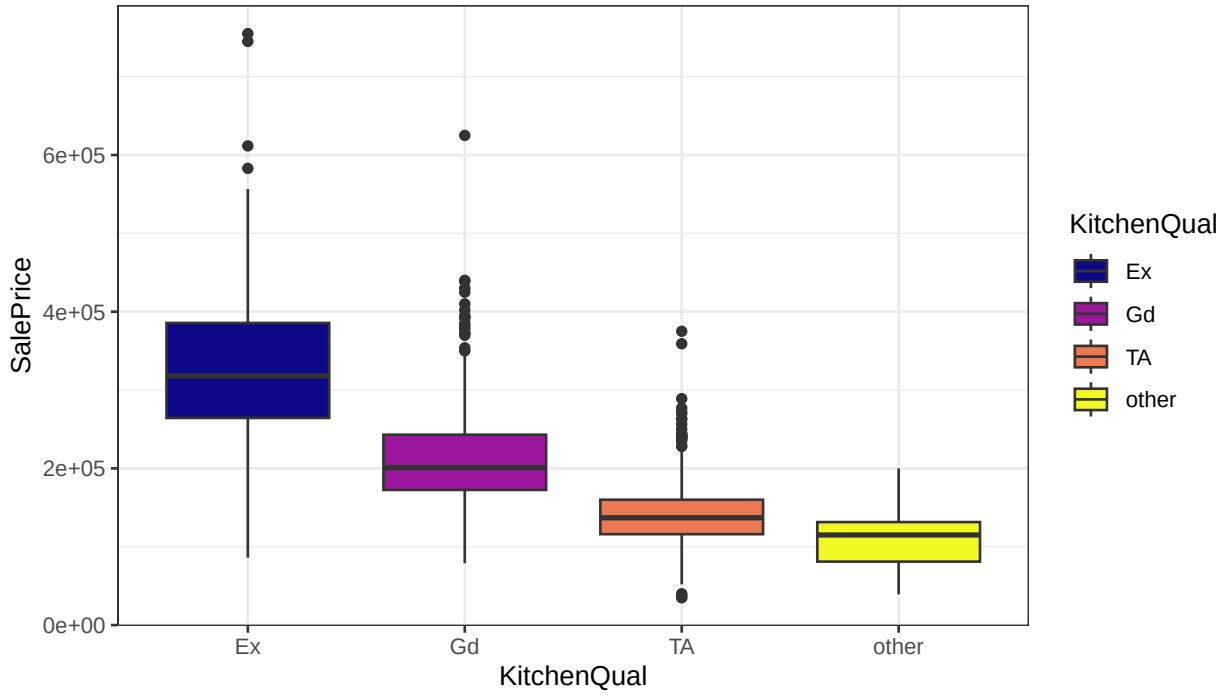

% of the variance explained by the PCs
ExplainedVar
0.20
0.15
0.10
0.05
0.00
0
10
20
30
PC
PC2
PC1
PC3
PC4
SalePrice
13.5
13.0
12.5
12.0
11.5
11.0
10.5

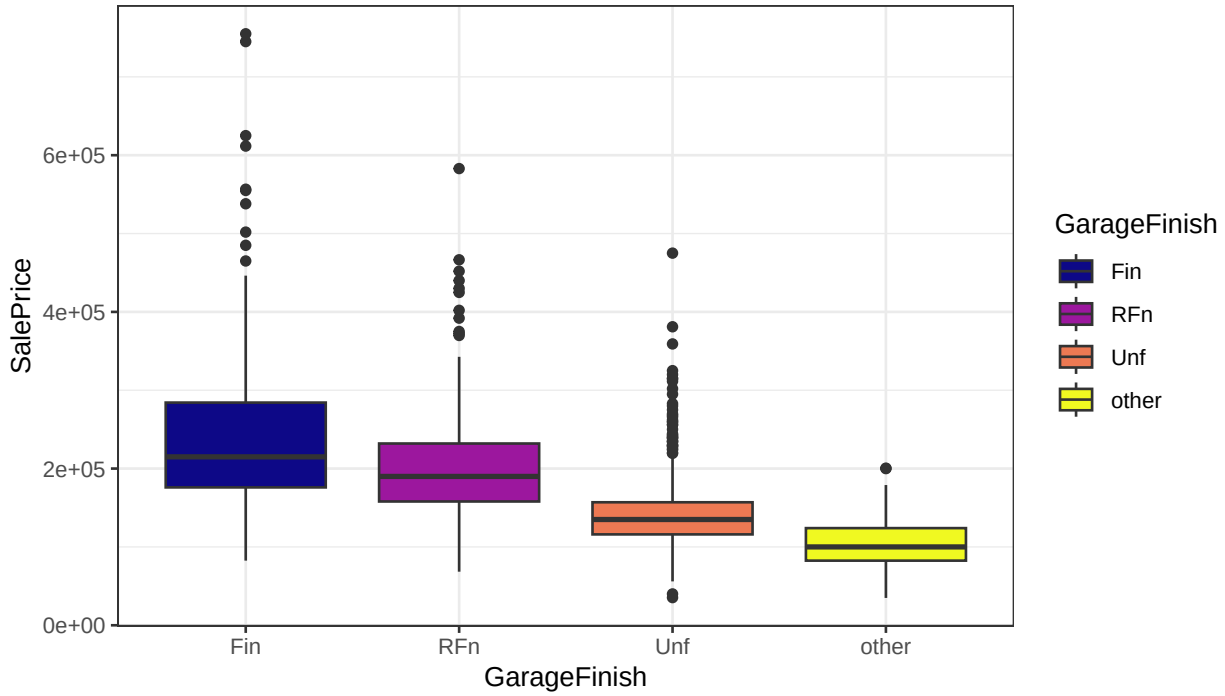


We concluded the exploratory analysis of the data by taking a look at the categorical variables. We plotted the boxplot of the Sale Price conditioned to every class of the categorical variables, looking for significant differences among them. We show here a few of the plots, concerning the most interesting covariates.

The last plot represents the mutual information matrix for the categorical covariates. It conveys information about the dependence between two variables, and we adopted it as a way of assessing correlation between these variables.


SalePrice
6e+05
4e+05
2e+05
0e+00
Ex
Gd
TA
other
BsmtQual
BsmtQual
Ex
Gd
TA
other


SalePrice
6e+05
4e+05
2e+05
0e+00
Ex
Gd
TA
other
KitchenQual
KitchenQual
Ex
Gd
TA
other



Linear Model

In this paragraph we discuss the building process of the linear model we employed to predict the response variable `SalePrice`.

The goal was to obtain a model showcasing a good trade-off between performance and complexity. Since the dataset we had at disposal consisted of many different variables, the first task was to select among them the covariates to use for our model, by choosing the ones which would lead to better performances in terms of explained variability of the response variable (higher *adjusted* R^2). Moreover, we wanted the vast majority of the terms to be statistically significant and therefore, we payed attention to avoid multicollinearity problems. Finally, in order to address the model's complexity, interpretability and to avoid overfitting, we simplified as much as possible the model's structure, while maintaining consistent results in terms of *adjusted* R^2.

In order to reduce from the beginning the risk of having multicollinearity problems, we first decided to remove from the dataset some variables which were highly correlated to/dependent from other variables; among a set of correlated variables, we kept in the dataset the one which was the most correlated to the response variable and therefore more useful to predict it. We then selected the covariates to remove by looking at the correlation matrix (for the numerical variables) and the mutual information matrix (for the categorical variables).

Starting from a linear model containing all the remaining variables in the dataset as covariates (*adjusted* $R^2 = 0.8683$), we used the `stepAIC` function (part of the `MASS` package) to select the linear model composed by the best combination of covariates with respect to the *AIC* (Akaike Information Criterion).

The AIC is a measure that balances the goodness of fit of a model with its complexity, penalizing the models that are too complex: better fits correspond to smaller AIC values.

The `stepAIC` function systematically adds or removes predictor variables from the model to find the combination that minimizes the AIC. The process involves iteratively fitting models, assessing their AIC, and deciding whether to add or remove variables based on the AIC improvement.

The `stepAIC` function returned the model which minimized the AIC: this model consisted of 30 covariates, so it was way less complex than the starting one; it showcased a good performance as well, since it had

adjusted $R^2 = 0.8696$. We also computed the *variance inflation factor* (VIF) associated to each predictor to spot eventual collinearity and we obtained $VIF_j < 4 \ \forall j$ (way below the critical value 10). Finally, the ANOVA suggested to keep this simpler model instead of the initial more complex one.

Indeed, we obtained a list of 30 selected variables to use as covariates of our linear model. In order to simplify the model we got using the `stepAIC` function, we then tried to fit a LASSO model using these variables, to see if we could remove some more predictors by shrinking their coefficients to zero. In this way we were able to exclude four more variables from our list of covariates; we then again used the `stepAIC` function starting from a linear model containing the remaining selected variables as predictors and obtained the following final linear model.

```
##
## Call:
## lm(formula = SalePrice ~ LotArea + LandSlope + Condition1 + BldgType +
##     OverallQual + YearBuilt + YearRemodAdd + RoofStyle + BsmtExposure +
##     BsmtFinSF1 + BsmtFinSF2 + TotalBsmtSF + HeatingQC + LowQualFinSF +
##     GrLivArea + BsmtFullBath + BedroomAbvGr + KitchenAbvGr +
##     KitchenQual + Functional + Fireplaces + GarageArea + WoodDeckSF +
##     ScreenPorch + PoolArea, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -394216  -13448    -705   12688  223596
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.019e+06  1.192e+05  -8.546  < 2e-16 ***
## LotArea            1.154e+04  2.248e+03   5.134 3.23e-07 ***
## LandSlopeMod       7.377e+03  3.934e+03   1.875 0.060988 .
## LandSlopeother    -9.169e+03  9.136e+03  -1.004 0.315717
## Condition1Norm     7.023e+03  3.503e+03   2.005 0.045169 *
## Condition1other   -8.242e+03  4.343e+03  -1.898 0.057933 .
## BldgTypeTwnhsE    -9.692e+03  3.613e+03  -2.683 0.007389 **
## BldgTypeother     -7.238e+03  3.955e+03  -1.830 0.067441 .
## OverallQual        1.234e+04  1.046e+03  11.798  < 2e-16 ***
## YearBuilt          2.366e+02  4.013e+01   5.896 4.64e-09 ***
## YearRemodAdd       2.367e+02  5.631e+01   4.203 2.80e-05 ***
## RoofStyleHip       4.803e+03  2.089e+03   2.299 0.021666 *
## RoofStyleother    -2.144e+03  5.567e+03  -0.385 0.700220
## BsmtExposureNo    -3.029e+03  2.360e+03  -1.284 0.199517
## BsmtExposureother  1.168e+04  2.754e+03   4.241 2.37e-05 ***
## BsmtFinSF1         1.434e+03  3.385e+02   4.235 2.43e-05 ***
## BsmtFinSF2        -8.165e+02  4.428e+02  -1.844 0.065421 .
## TotalBsmtSF        2.250e+01  2.577e+00   8.728  < 2e-16 ***
## HeatingQCGd       -4.541e+03  2.390e+03  -1.900 0.057629 .
## HeatingQCTA       -6.873e+03  2.246e+03  -3.059 0.002260 **
## HeatingQCother    -3.223e+03  4.702e+03  -0.686 0.493073
## LowQualFinSF      -5.272e+01  1.674e+01  -3.149 0.001673 **
## GrLivArea          6.302e+01  2.820e+00  22.352  < 2e-16 ***
## BsmtFullBath       4.190e+03  1.967e+03   2.130 0.033373 *
## BedroomAbvGr      -6.839e+03  1.315e+03  -5.202 2.25e-07 ***
## KitchenAbvGr      -1.422e+04  4.915e+03  -2.892 0.003883 **
## KitchenQualGd     -4.603e+04  3.524e+03 -13.063  < 2e-16 ***
## KitchenQualTA     -4.702e+04  4.153e+03 -11.322  < 2e-16 ***
## KitchenQualother  -3.809e+04  6.666e+03  -5.714 1.34e-08 ***
```

```
## FunctionalTyp      1.581e+04  5.301e+03   2.982 0.002913 **
## Functionalother   -4.602e+03  6.335e+03  -0.726 0.467753
## Fireplaces         3.439e+03  1.481e+03   2.322 0.020396 *
## GarageArea         2.783e+01  4.969e+00   5.600 2.57e-08 ***
## WoodDeckSF         1.879e+01  6.783e+00   2.769 0.005690 **
## ScreenPorch        2.910e+01  1.448e+01   2.010 0.044656 *
## PoolArea           6.887e+01  2.080e+01   3.311 0.000954 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29470 on 1422 degrees of freedom
## Multiple R-squared:  0.8659, Adjusted R-squared:  0.8626
## F-statistic: 262.2 on 35 and 1422 DF,  p-value: < 2.2e-16
```

The variables we selected as covariates for the final linear model we developed are then the following ones:

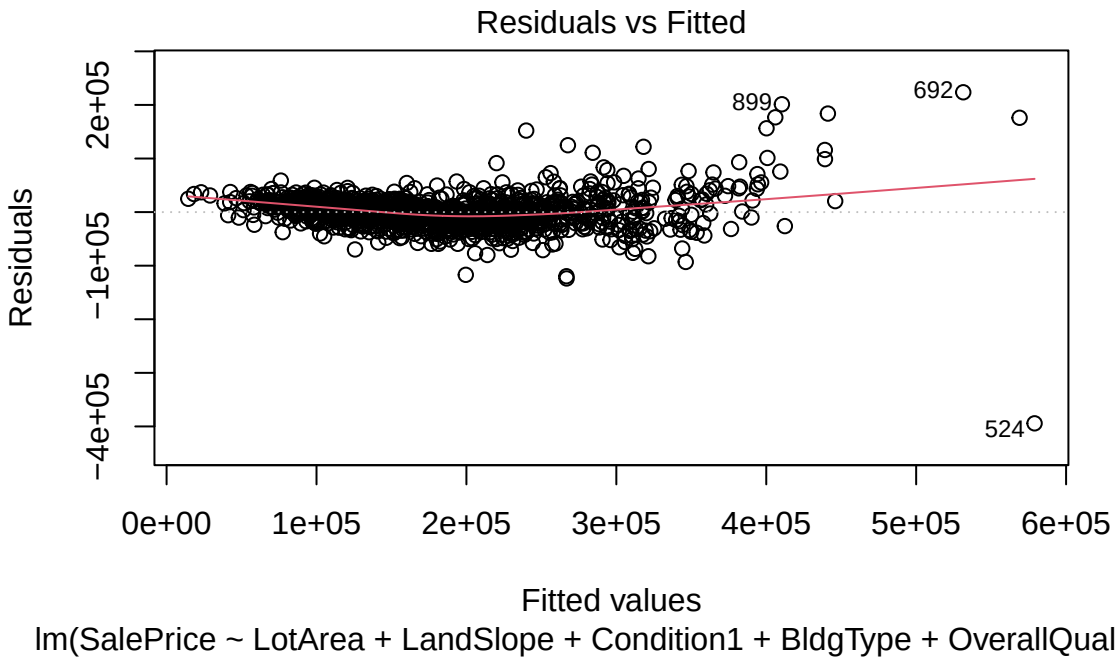
```
##  [1] "LotArea"      "LandSlope"    "Condition1"   "BldgType"     "OverallQual"
##  [6] "YearBuilt"    "YearRemodAdd" "RoofStyle"    "BsmtExposure" "BsmtFinSF1"
## [11] "BsmtFinSF2"   "TotalBsmtSF"  "HeatingQC"    "LowQualFinSF" "GrLivArea"
## [16] "BsmtFullBath" "BedroomAbvGr" "KitchenAbvGr" "KitchenQual"  "Functional"
## [21] "Fireplaces"   "GarageArea"   "WoodDeckSF"   "ScreenPorch"  "PoolArea"
```

We can underline the fact that some of these variables are the ones we noticed to contribute noticeably to the first component in the *dimensionality reduction* paragraph.

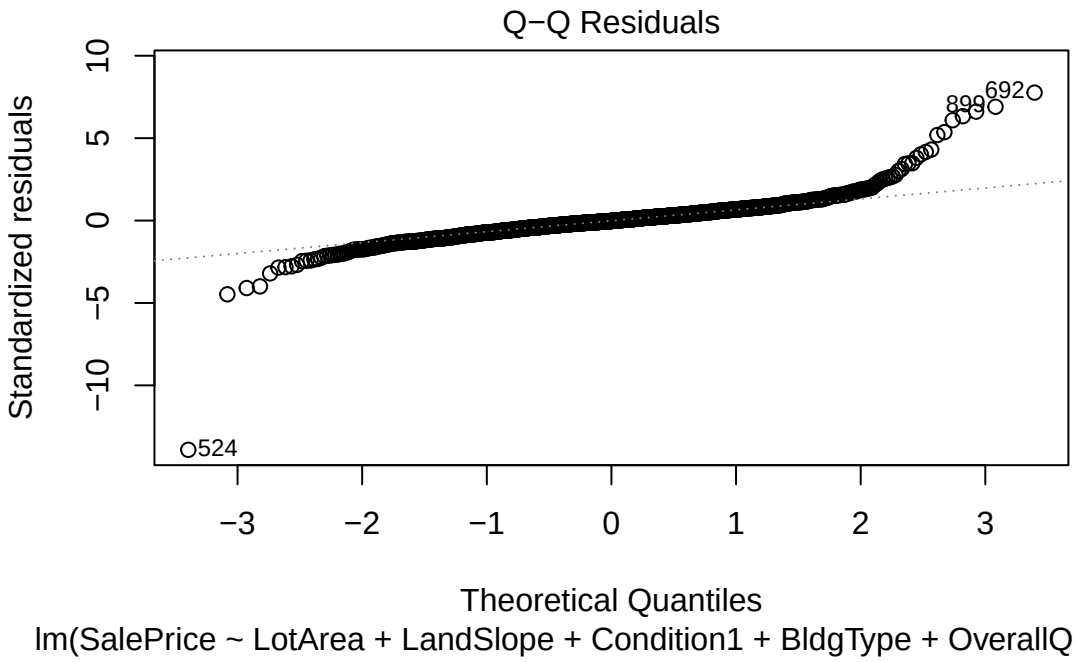
This liner model has 25 predictors and *adjusted* $R^2 = 0.8626$; moreover, by looking at the p-values in the model's summary, we can see that the terms are in general statistically significant. We also computed the *variance inflation factor* (VIF) associated to each predictor to spot eventual collinearity and we obtained $VIF_j < 4 \; \forall j$.

Following this procedure, we therefore built a linear model which showcases a good trade-off between its performance and its complexity:

- its covariates are selected to minimize the AIC;
- its performance is good in terms of *adjusted* R^2 (moreover, its R^2 is not far from the complete model's one, which was approximately equal to 0.88);
- it does not have multicollinearity problems;
- we were able to simplify the model's structure by reducing the number of predictors, while maintaining consistent results in terms of the model's performance.



By plotting the residuals against the fitted values, we can see that the linearity and the homeschedasticity assumptions are overall met; however, we can spot some outliers depending on the dataset. Moreover, we can notice that we obtain worse results as the fitted values increase: this could be due to the fact that our dataset (and therefore our model) don't contain variables which are useful to predict the prices of more expensive houses.



By looking at the qqplot, the normality assumption could be questioned; as already mentioned, we tried to fit the model using the logarithm of the response variable to fix this problem, but it didn't improve the situation.

We decided to not include interaction terms in our linear model, since they would have decreased its interpretability, increased its complexity (and hence the risk of overfitting and multicollinearity) and its performance was already satisfying.

Random Forest

```
##
## Call:
##  randomForest(formula = SalePrice ~ ., data = df, ntree = 500,      importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 22
##
##           Mean of squared residuals: 778408109
##                     % Var explained: 87.67
```

The "mean of squared residuals" is the MSE of the out-of-bag errors.

In Random Forest, out-of-bag (OOB) observations refer to the data points that are not included in the bootstrap sample used to grow a particular tree in the forest. Bootstrap sampling involves randomly selecting data points from the original dataset with replacement, which means some observations may be selected multiple times while others may not be selected at all.

Since each tree in a Random Forest is grown using a bootstrap sample, there will be some data points that were not included in the sample used to build that specific tree. These out-of-bag observations are essentially left out during the training process of that tree.

After growing each tree in the forest, the out-of-bag observations can be used to evaluate the performance of that tree by calculating prediction errors. This process is repeated for each tree, and the aggregate prediction errors across all trees are used to assess the overall performance of the Random Forest model.

The mean of squared residuals obtained is a good result considering the scale of houses prices.

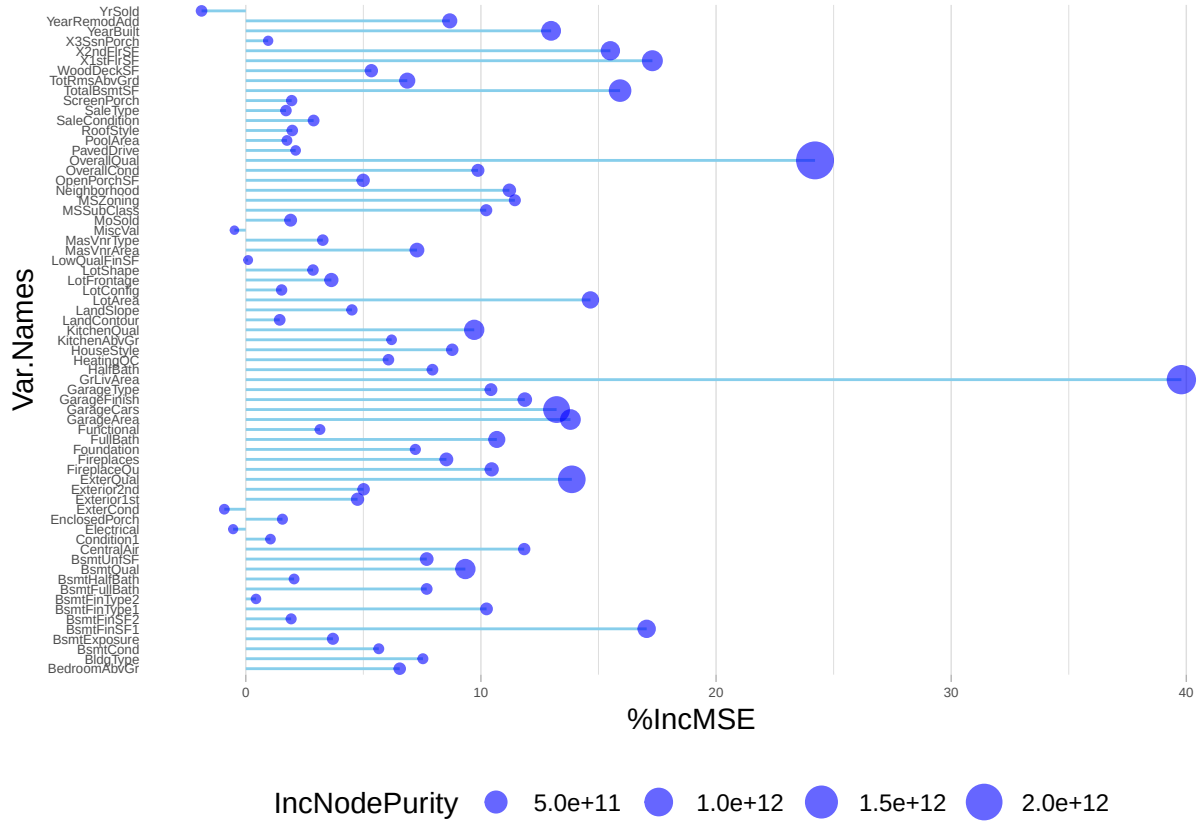
If we compute the root of the mean of MSE of single trees we obtain 2.8557113×10^4 that is a reasonable value, also considering other models.

The "% Var explained" term is a "pseudo R-squared", computed as `1 - MSE/Var(y)`. This value represents the proportion of the total variance in the response variable that is accounted for by the predictor variables included in the Random Forest model.

The "% Var explained" term is a useful metric for assessing the goodness-of-fit of the Random Forest model. A higher percentage indicates that a larger proportion of the variability in the response variable is captured by the predictors included in the model, suggesting a better fit.

Our result suggests that Random Forest model can explain pretty well the total variance of the response variable.

We can compare this result with the Adjusted R-squared obtained with the linear model.



This graph represents %IncMSE and IncNodePurity, usefull to visualize variables' importance.

"%IncMSE" is the most robust and informative measure. It is the increase in MSE of predictions (estimated with out-of-bag-CV) as a result of variable j being permuted (values randomly shuffled). The higher, the more important.

"IncNodePurity" relates to the loss function which by best splits are chosen. The loss function is MSE for regression and GINI-impurity for classification. More useful variables achieve higher increases in node purities, that is to find a split which has a high inter node 'variance' and a small intra node 'variance'.

Inter-node variance refers to the variance between different nodes or branches of a decision tree or ensemble. In a Random Forest, inter-node variance specifically measures the variability in predictions across different trees in the forest.

Intra-node variance refers to the variance within a specific node or branch of a decision tree or ensemble. It quantifies the variability in predictions or target variable values within a particular subset of data represented by a node.

In summary, while inter-node variance focuses on the variability between different nodes or trees in an ensemble, intra-node variance focuses on the variability within individual nodes or trees.

As we can see by the graph, a lot of variables selected with the analysis using linear models are the most important ones also for the Random Forest model.

GAM Model

We decided to improve the simple linear model using a Generalized Additive Model, which allows for more complex interactions between the covariates and the sale price. We started by taking the linear model and added smoothing splines to the variables in the model for which a linear fit did not seem appropriate. We kept the ones which resulted statistically significant and contributed to increase the adjusted R^2 of the model. We then proceeded to do the same with the variable that previously did not explain the response variable, and decided whether to include them basing our decision on the outcome of an ANOVA test. With this procedure we manage to obtain an R^2 above 0.9, which is a noticeable improvement from the linear model.

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## SalePrice ~ LotArea + LandSlope + Condition1 + BldgType + s(OverallQual) +
##     s(YearBuilt) + s(YearRemodAdd) + RoofStyle + BsmtExposure +
##     s(BsmtFinSF1) + BsmtFinSF2 + TotalBsmtSF + HeatingQC + LowQualFinSF +
##     s(TotRmsAbvGrd) + GrLivArea + BedroomAbvGr + KitchenQual +
##     Functional + Fireplaces + s(GarageArea) + s(ScreenPorch) +
##     s(MasVnrArea) + s(X1stFlrSF) + s(X2ndFlrSF) + BsmtQual +
##     CentralAir
##
## Parametric coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -21442.882   7310.893  -2.933  0.00341 **
## LotArea            13310.068   1965.837   6.771 1.89e-11 ***
## LandSlopeMod        7596.648   3368.277   2.255  0.02427 *
## LandSlopeother     -4033.782   7873.092  -0.512  0.60849
## Condition1Norm      9301.849   2971.621   3.130  0.00178 **
## Condition1other    -4258.154   3704.779  -1.149  0.25060
## BldgTypeTwnhsE     -4795.777   3174.799  -1.511  0.13113
## BldgTypeother     -11753.563   2824.610  -4.161 3.36e-05 ***
## RoofStyleHip        1610.973   1879.675   0.857  0.39157
## RoofStyleother      3400.541   4752.695   0.715  0.47442
## BsmtExposureNo     -2164.106   2041.647  -1.060  0.28934
## BsmtExposureother   5812.138   2414.068   2.408  0.01619 *
## BsmtFinSF2           430.192    417.801   1.030  0.30335
## TotalBsmtSF           19.497      3.682   5.295 1.38e-07 ***
## HeatingQCGd        -3604.592   2082.669  -1.731  0.08372 .
```

```
## HeatingQCTA        -3931.578   2055.033  -1.913  0.05594 .
## HeatingQCother     -2151.769   4157.976  -0.518  0.60489
## LowQualFinSF         -45.269     19.146  -2.364  0.01820 *
## GrLivArea             52.024     12.359   4.209 2.73e-05 ***
## BedroomAbvGr       -4235.681   1324.262  -3.199  0.00141 **
## KitchenQualGd     -14854.008   3492.900  -4.253 2.26e-05 ***
## KitchenQualTA     -21436.835   3902.466  -5.493 4.70e-08 ***
## KitchenQualother  -17625.466   5899.951  -2.987  0.00286 **
## FunctionalTyp      11857.074   4599.097   2.578  0.01004 *
## Functionalother    -6742.192   5439.497  -1.239  0.21538
## Fireplaces          5692.181   1322.706   4.303 1.80e-05 ***
## BsmtQualGd        -13728.531   3364.905  -4.080 4.76e-05 ***
## BsmtQualTA        -18067.817   4156.608  -4.347 1.48e-05 ***
## BsmtQualother     -10990.516   5599.905  -1.963  0.04989 *
## CentralAirY         6974.792   3407.559   2.047  0.04086 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                    edf Ref.df      F p-value
## s(OverallQual)  3.564  4.587 45.448 < 2e-16 ***
## s(YearBuilt)    2.669  3.396 13.125 < 2e-16 ***
## s(YearRemodAdd) 8.047  8.747  6.646 < 2e-16 ***
## s(BsmtFinSF1)   8.458  8.915 13.540 < 2e-16 ***
## s(TotRmsAbvGrd) 3.011  3.849  4.282 0.00271 **
## s(GarageArea)   8.312  8.850  6.311 < 2e-16 ***
## s(ScreenPorch)  5.106  6.160  3.519 0.00165 **
## s(MasVnrArea)   2.708  3.290  3.184 0.02141 *
## s(X1stFlrSF)    6.872  7.969 12.917 < 2e-16 ***
## s(X2ndFlrSF)    8.151  8.746 12.944 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 119/120
## R-sq.(adj) =  0.904   Deviance explained =   91%
## GCV = 6.4475e+08  Scale est. = 6.0671e+08  n = 1458
```

Neural Network Model

Our final model is a Neural Network, known for its flexibility and ability to capture powerful non-linear relationships between inputs and outputs, though at the cost of explainability.

The network is structured in layers, with the input layer receiving the input, and subsequent layers processing the output of the previous one. Coefficients are learned through stochastic gradient descent, and each layer applies a scalar non-linear function before passing the output to the next layer.

Our specific model is a small-sized multilayer perceptron (MLP) with three fully connected layers. After some architecture tuning, we set the first and second layers to have 32 neurons each, while the last layer, predicting the scalar output variable, consists of a single neuron. In order to stabilize training we normalize all numerical features to lie in the $[0, 1]$ range. The categorical variables on the other hand are one-hot encoded before feeding to the network. All layers use the sigmoid (inverse logit) non linear activation function.

Instead of feeding all features in the dataset to the NN, we find that using the same variables picked for the GAM model leads to consistently better results.

Here's the architecture of the model:

```
## Model: "sequential"
## _________________________________________________________________________
##  Layer (type)                 Output Shape              Param #    Trainable
## =========================================================================
##  dense_2 (Dense)              (None, 32)                1600       Y
##  batch_normalization_1 (Batch  (None, 32)               128        Y
##  Normalization)
##  activation_1 (Activation)    (None, 32)                0          Y
##  dense_1 (Dense)              (None, 32)                1056       Y
##  batch_normalization (BatchNo  (None, 32)               128        Y
##  rmalization)
##  activation (Activation)      (None, 32)                0          Y
##  dense (Dense)                (None, 1)                 33         Y
## =========================================================================
## Total params: 2945 (11.50 KB)
## Trainable params: 2817 (11.00 KB)
## Non-trainable params: 128 (512.00 Byte)
## _________________________________________________________________________
```

The batch normalization layers that appear in the summary are commonly used to stabilize and speed up training [2]. They perform sample wise centering and scaling of the input of each layer.

We can see from the summary that the model has a total of 2817 trainable parameters. As stated before the training is performed using SGD, in particular we use the Adam optimizer which is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. As in other gradient descent methods we have to pick a learning rate, in our case we fix it to $1e-4$ and decrease it during training if the validation loss does not improve for a certain amount of epochs. As a loss function we use the Mean Squared Error (MSE).

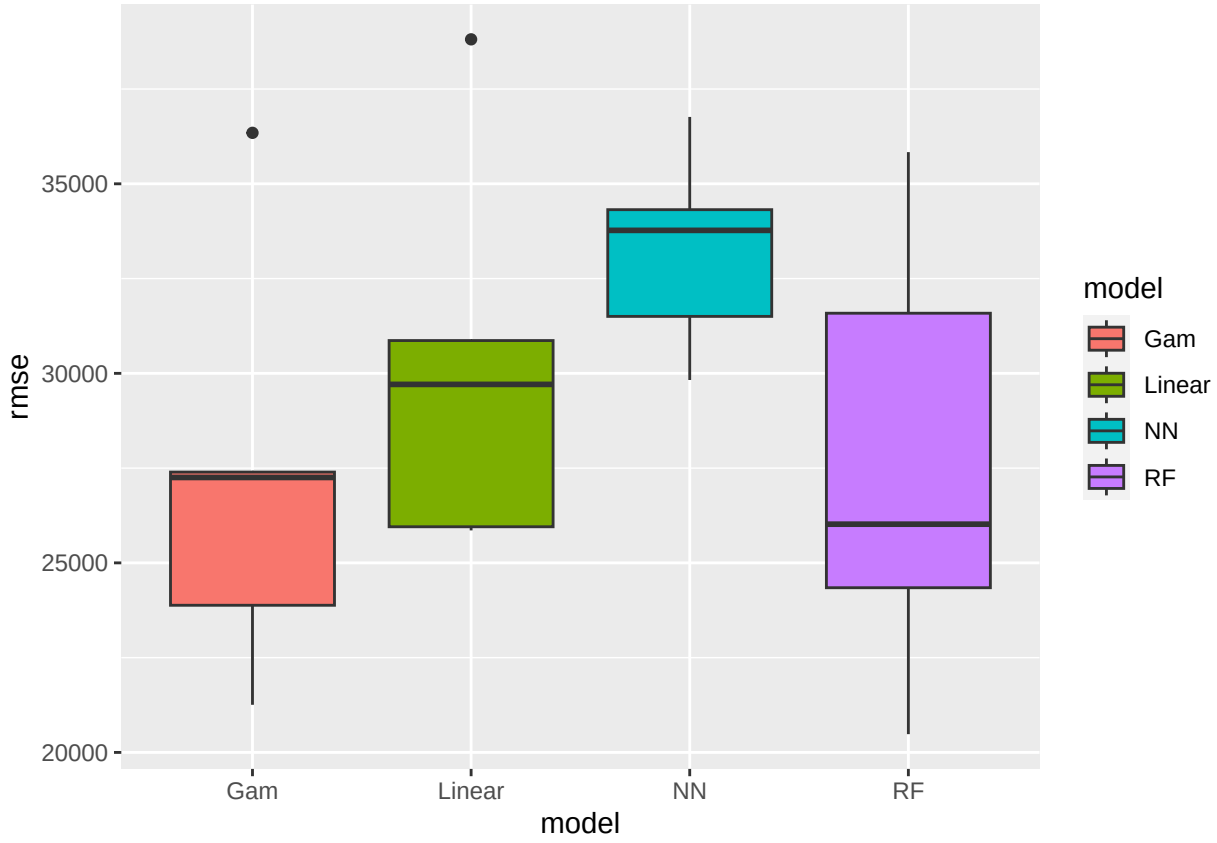
Conclusions

Having not utilized a separate test dataset since we weren't provided with one, we conducted the models' assessment phase using the cross-validation (CV) technique. The chosen CV method employs 10 folds, thus utilizing at each iteration 1314 observations for training and 146 for testing. We selected the RMSE as the assessment metric.

Comparing the models, as evident from the graph, the Random Forest model outperforms the others. Despite the comparable median of the GAM to that of the Random Forest, the former exhibits higher variance. Conversely, the linear model, although with lower variance, presents a higher median compared to both GAM and Random Forest. The Neural Network, on the other hand, yields unsatisfactory results, probably because of the requirement of a larger amount of data than what was available in our dataset.

In conclusion, the Random Forest model emerges as the superior model in terms of RMSE among those analyzed.

Another factor for evaluation is the explainability of the models. The linear model undoubtedly stands out in terms of explainability, followed by the GAM model. Conversely, the Random Forest model and above all the neural network are more challenging to visualize.



References

1. Anna Montoya D. House prices - advanced regression techniques. 2016.
2. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pmlr; 2015. p. 448–56.