Application of High Dimensional Techniques to House Price Prediction

Pala, Alessandro
alessandro.pala@studenti.unipd.it

Tremaggi, Domenico
domenico.tremaggi@studenti.unipd.it

Abstract

We explore regression and regularization techniques to predict house prices based on physical, qualitative, locational, and area features. The best result was achieved with a Generalized Additive Model (GAM), explaining approximately 85% of the variance. Other methods, including linear regression, Poisson regression with Elastic Net, and Group Lasso, were tested, with similar performance. Results indicate that house prices respond highly and independently to house features, and that such relationship can be both linear or polynomial.

1 Introduction

In order to approach the dataset, we first think about which models and variables can predict real estate prices in the real world, with the constraint of the set of variables given by the dataset. Since we handle a price prediction regression, a fair comparison of linear and non-linear basic models - be them regularized or not - would be multivariate. Hence, we first carry a synthetic application of the following models: multivariate linear model and multivariate Poisson regression with an $\alpha = 0.5$ Elastic Net.

In order to understand the higher bound of possible model performance on the dataset, we train a deep multi-layer linear perceptron and we regularize it with a simple Lasso. Then, we utilise a Group Lasso regression on two sets of groups of variables that we think may be good predictors of house prices in the real world. The first set of groups uses variables that are usually clustered together by real estate professionals when they do house price estimation. The second set of groups uses clusters of highly statistically correlated (Pearson's r) variables(Figure **??**). Finally, we run a generalized additive model allowing for up to four degrees of freedom on each numerical variable and compare all results.

Selection for regularized models is carried through with cross-validation.

2 Dataset

The dataset contains real estate property listings, each described by a variety of attributes. Temporal attributes are: the listing date, the year the property was built, and the year of its last renovation. Physical characteristics are: the number of bedrooms and bathrooms, living area size, total lot size, number of floors, basement area, and living area above ground. Scenic qualities are represented by a waterfront view indicator, a view quality rating, an overall condition rating, and an overall grade rating. Locational data instead include zip code, latitude, and longitude. There is also some additional spatial context given by the average living area and lot sizes of the 15 nearest properties.

2.1 Pre-processing

In order to pre-process the data, we first convert the date column into a proper date object by extracting the first 8 characters (in "YYYYMMDD" format). We then replace 0 values in the "yr_renovated" column with NA, treating them as missing data. Duplicate rows are removed based on the id column. We add new variables in the dataset: total_sqft calculates total area by summing living and basement space, bath_per_bed computes the bathroom-to-bedroom ratio (defaulting to 0 if bedrooms are 0), and total_rooms sums bedrooms and bathrooms. sqft_diff_15 measures the difference between the property's living area and that of its 15 nearest neighbors, while age_since_reno calculates the property's age since construction, defaulting to 0 if the year is invalid. (Figure **??**) The first, second, fifteenth and sixteenth columns are dropped since they are analytically irrelevant (date, id, with yr_built and yr_renovated being substituted by "age"). We then

scale the features of the data frame. The price and the non-structural features are extracted as columns 2 to 18 and scaled in the new matrix X, which will be used for the Group Lasso. The price, or target, is stored in a separate vector Y.

3 Base Models

3.1 Multivariate linear model

Multivariate Linear Regression is an extension of linear regression where the goal is to model the relationship between multiple independent variables (also called predictors or features) and a dependent variable (also called the target or response variable). It is used when you have more than one predictor variable that you believe influences the dependent variable.

Mathematical equation

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \epsilon \quad (1)$$

where:Y is the dependent variable, $(X_1, X_2, \ldots, X_k)$ are the independent variables, β_0 is the intercept, $(\beta_1, \beta_2, \ldots, \beta_k)$ are the coefficients and ϵ is the error term.

Experiments

In our case of interest we use all the the available variables to estimate the house prices, including the artificial ones. The model explains approximately 69% of the variability in the response variable (Figure **??**) with many variables exhibiting p-values ¡ 0.001 and, notably, **sqft_living**, **grade**, and **waterfront** having particularly high coefficients (Table **??**). Other metrics, along with the plots (Figure **??**), tell us that there's some unwanted moderate-scale variability in the predictions.

3.2 Multivariate GLM with Elastic Net

The Poisson Distribution

Poisson regression is based on the Poisson distribution, which describes the probability of a given number of events occurring in a fixed interval of time or space, given the average rate of occurrence.
The probability mass function of a Poisson-distributed random variable Y is:

$$P(Y = y) = \frac{\lambda^y e^{-\lambda}}{y!} \quad (2)$$

Where: y is the count of events (e.g., number of accidents, number of emails received) and λ is the rate of occurrence, which is the expected number of events in the fixed interval.

The Poisson Regression Model

Poisson regression assumes that the logarithm of the expected count (rate) is a linear function of the independent variables.
The model is expressed as:

$$\log(\lambda_i) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \quad (3)$$

where: $(X_1, X_2, \ldots, X_k)$ are the indipendent variables, β_0 is the intercept, $(\beta_1, \beta_2, \ldots, \beta_k)$ are the regression coefficients.

Elastic Net

In regular linear regression or GLMs, the model can overfit the data if there are many predictors, especially when some predictors are irrelevant or highly correlated, but it can also use some sparsity enforcement, so we applied an Elastic Net regression. This kind of regularization combines L_1 and L_2 regularizations and applies a penalty to coefficients as to:

- Lasso (L_1): Shrink some coefficients to zero, performing feature selection (Figure 4).
- Ridge (L_2): Shrink the coefficients (but not set them to zero) helping with multi- collinearity, which our dataset suffers from.

Mathematically, the Elastic Net penalty is given by:

$$\text{Penalty} = \gamma \left[\alpha \sum_{j=1}^{k} |\beta_j| + \frac{1}{2}(1-\alpha) \sum_{j=1}^{k} \beta_j^2 \right] \quad (4)$$

where γ controls the strength of the regularization (larger γ means more regularization), and α controls the balance between Lasso and Ridge.

Poisson Regression with Elastic Net

When you combine Poisson regression with Elastic Net regularization the model becomes:

$$\log(\lambda_i) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \\ + \gamma \left[\alpha \sum_{j=1}^{k} |\beta_j| + \frac{1}{2}(1-\alpha) \sum_{j=1}^{k} \beta_j^2 \right] \quad (5)$$

This means that the objective is to minimize the following loss function:

$$\text{Loss} = \sum_{i=1}^{n} (-Y_i \log(\lambda_i) + \lambda_i) +$$

$$+\gamma \left[\alpha \sum_{j=1}^{k} |\beta_j| + \frac{1}{2}(1-\alpha) \sum_{j=1}^{k} \beta_j^2 \right]$$

Experiments

The Poisson regression model had a smooth regularization (Figure **??**) and successfully performed variable selection (Table **??**) with a relatively normal-looking cross validation plot which resulted in a very large best-lambda value (Figure **??**). However, this model's performance was slightly lower than the linear model, explaining 67.5% of the variance, which means either a wrong distributional assumption or linearity in the relationship in the data (Figure **??**).

4 Group Lasso

The Group Lasso is an extension of the classical Lasso method applied for regularization and variable selection designed to work with grouped variables, and such a situation is helpful when given variables naturally fall into groups, and we want to choose or discard entire groups of variables simultaneously rather than choosing or dropping individual variables.

The main mathematical goal of the Group Lasso is to do regression by penalizing the sum of the norms of the coefficients in each group. That will induce sparsity at the level of groups, that is, all coefficients in a group will either be shrunk towards zero, or not. Therefore, entire groups of variables are either selected or not.

4.1 Mathematical Formulation

For a given linear regression problem, the model can be represented as:

$$y = X\beta + \epsilon \tag{6}$$

where: y is the $n \times 1$ vector of observed responses, X is the $n \times p$ matrix of predictors, β is the $p \times 1$ vector of regression coefficient and ϵ is the error term, assumed to be normally distributed.

In ordinary linear regression, we try to minimize the residual squared error:

$$\min_{\beta} \left(\frac{1}{2n} \|y - X\beta\|_2^2 \right) \tag{7}$$

In the case of Group Lasso, we add a regularization term to the objective function that penalizes the coefficients in groups. If the variables are divided into G groups, and group g contains p_g variables, the Group Lasso penalty is given by:

$$\lambda \sum_{g=1}^{G} \|\beta_g\|_2 \tag{8}$$

The idea would be: where: β_g represents the coefficient corresponding to the g-th group of variables, $\|\beta_g\|_2$ is the euclidean norm of the coefficient vector, λ is a regularization parameter that regulates its strength.

Thus, the optimization problem is:

$$\min_{\beta} \left(\frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \sum_{g=1}^{G} \|\beta_g\|_2 \right) \tag{9}$$

4.2 Experiments

The model performeds almost identically (Figure **??**, **??**), and as well as the linear regression, explaining about 60% of the variability in price. The coefficient path plot shows one easily regularizable fast-descending line, with the others being fairly constant throught lambda (Figure **??**), a result which we also see if we extract each group's contribution in explaining price after cross validation: in the cross-validation lambda choice (Figure **??**), no group was brought to zero, even though there are significant differences in group importance for both models, which also says something about the adjusted R-squared. Each set's most contributing group was almost as doubly important as the others (Figure **??**), so it makes sense to mention them: for the first model (real-life modeling) the most important group was group 1, consisting of physical characteristics; bedrooms, bathrooms, lot size and floors. For the second model (inter-correlated features) the most important group was group 4, consisting of house quality measures and extra features; grade, condition, view, waterfront.

5 Generalized Additive Model

5.1 Mathematical Formulation

Generalized Additive Models (GAMs) are a flexible class of models that extend GLMs by allowing the relationship between the predictors and the response to be modeled as a sum of smooth, potentially nonlinear functions. The mathematical formulation of a GAM is as follows:

We assume an additive relationship to extend the basis function:

$$f(x) = \sum_{j=1}^{p} f_j(x_j),$$

Each $f_j(x_j)$ may be estimated via some scatterplot smoother S, such as regression splines, loess, parametric methods, etc.

The general form of the additive model is:

$$y = \beta + \sum_{j=1}^{p} f_j(x_j) + \phi,$$

where β is the intercept, and the functions $f_j(x_j)$ are estimated by the backfitting algorithm.

The backfitting algorithm estimates $f_j(x_j)$ iteratively:

1. Initialize: $\hat{\beta} = \frac{\sum_i y_i}{n}, \hat{f}_j = 0$. 2. For each j, update $\hat{f}_j(x_j)$ using the smoother S:

$$\hat{f}_j(x_j) \leftarrow S\left(y - \hat{\beta} - \sum_{k \neq j} \hat{f}_k(x_k)\right).$$

The model is fitted by iteratively smoothing:

$$y = f_1(x_1) + \cdots + f_p(x_p) + \phi.$$

Each function $f_j(x_j)$ is updated in turn, based on the current estimates of the other functions.

5.2 Experiments

The firs result of the model is the distribution of the residuals, which exhibits a large range and confirms the presence of outliers in the data, with the IQR going from $-56,686$ to $52,102$ (Table **??**). The model is good at capturing the variance in the data, although and looking at the local scoring process we see a lack of iterations, which means model convergence with no need of further refinements.

The ANOVA table for parametric effects shows that most variables have statistically significant effects on price and, surprisingly, baths per bed (previously significant) being the only one not reaching a satisfactory p-value. The ANOVA table for nonparametric effects (Table **??**) shows similar results with all variables being important to predict the price.

Zooming out and looking at the model's general performance, we see that this is the most successful model, explaining circa 85% of the variability in price, and its MSE being less than half of the previous models' MSE (Figure **??**). By looking at the plots, we can notice that even though many features were best fitted with polynomially, most of them still retained linearity (Figure **??**). We also see that , just like in the other models, there's some heteroskedasticity, with models specifically struggling with very large values of price (which are also rare).

6 Neural Network

We finally train a deep neural network in order to have an estimate of the best possible result in any linear statistical context and, most of all, to understand if our choice of models was insufficient.
We kept the architecture neither too deep nor too shallow (Figure **??**) given the medium number of features. We see very clearly that the network uses the features of the dataset to predict extremely well (Figure **??**) and with no overfitting (Figure **??**). The goal of this small experiment was also to satiate any doubt about the dataset being "corrupted," and that any model could somehow suffer from problems. Indeed, the dataset is not corrupted.

7 Conclusion

As shown by the GAM, house prices are not static and respond highly to all sorts of house features, both physical, such as the number of bedrooms or the square feet of the living area, and locational, such as the zipcode or the neighborhood quality. Grouping such features either by mimicking real life statistical groups or according to their inter-correlation did not produce better results than simple models. The latter, if compared to the GAM, seem to highlight that they ignore some variables because they don't capture more complex nonlinear effects, captured instead by the GAM.

A Visualization

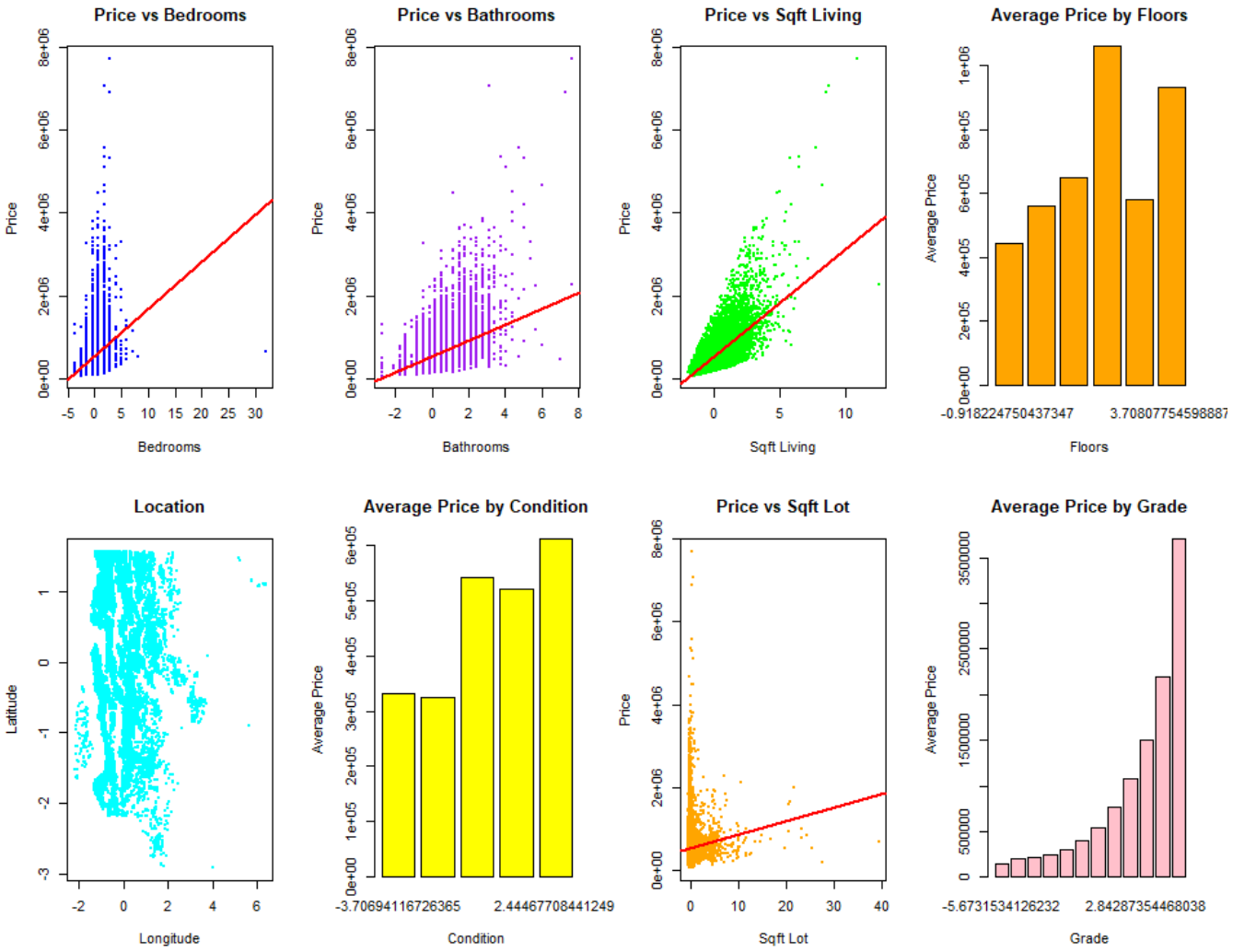

Figure 1: Some simple EDA

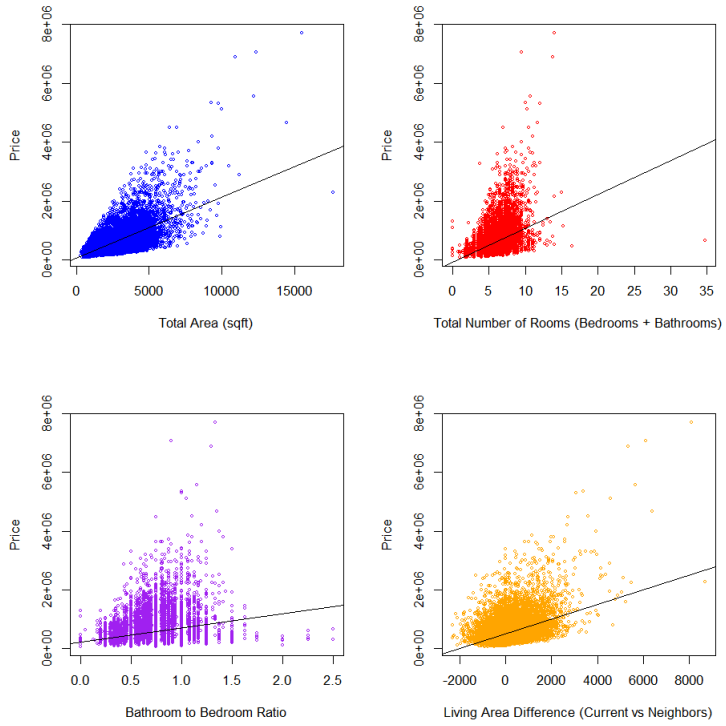

Figure 2: How price behaves with aggregated variables

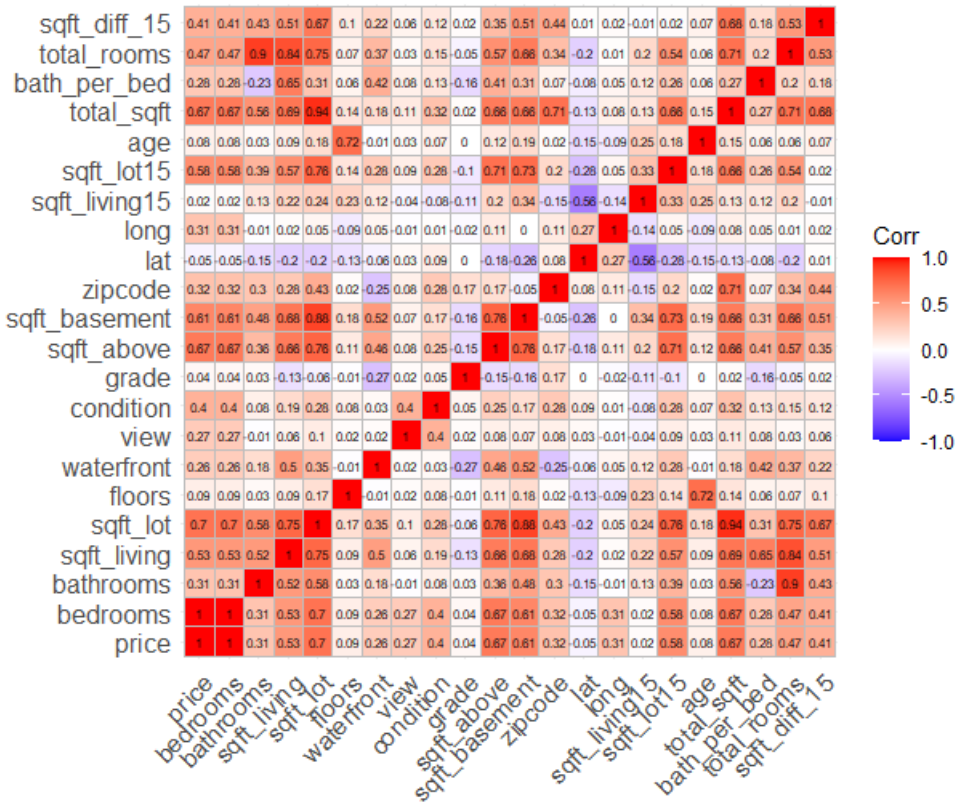

Figure 3: Correlation matrix to pick groups in the second set of Group Lasso

B Regressions

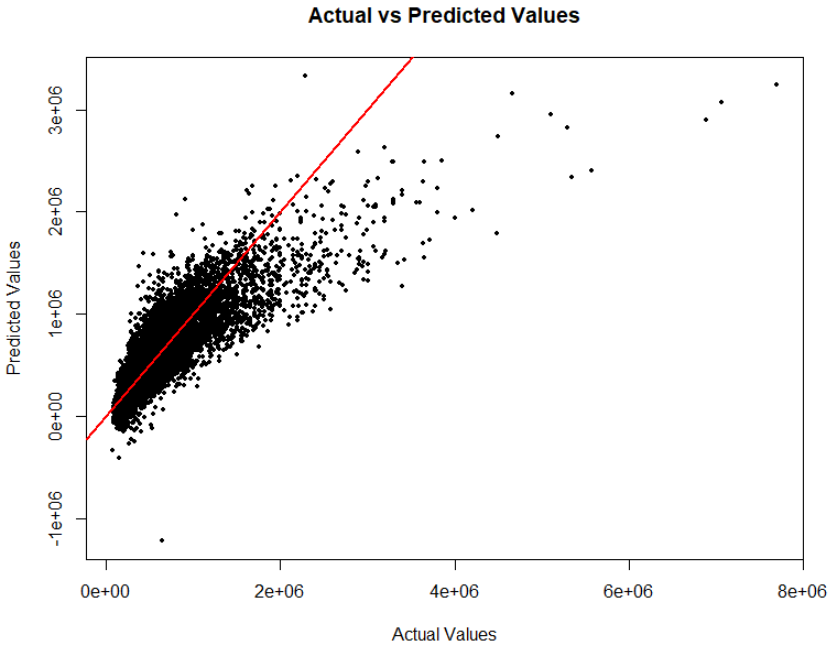

Figure 4: Linear regression actual vs predicted

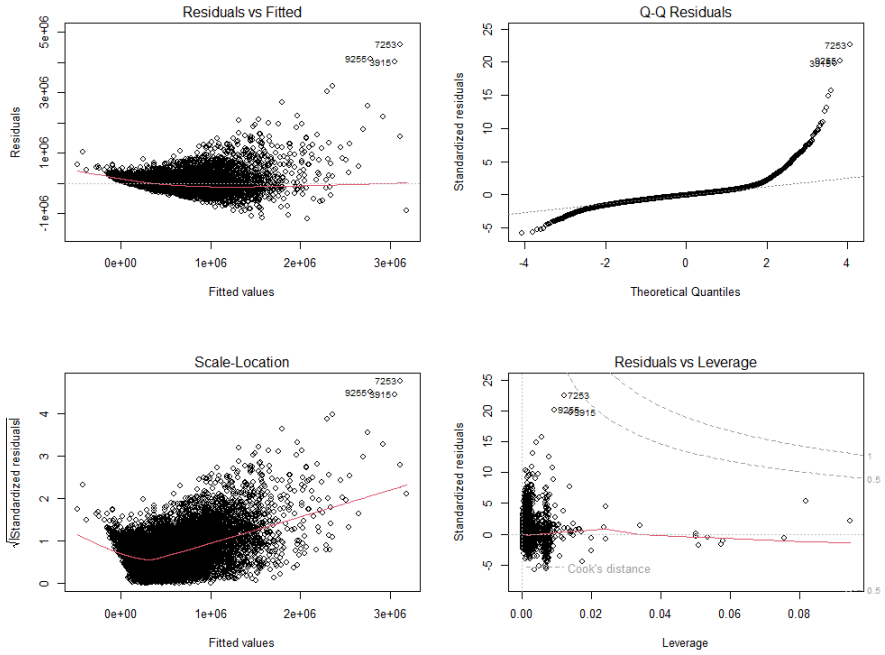

Figure 5: Multivariate linear regression plot

Variable	Estimate	Std. Error	t value	p value
(Intercept)	5.359×10^5	1.402×10^3	3.821×10^2	$< 2 \times 10^{-16}$ ***
bedrooms	-2.994×10^4	1.819×10^3	-1.647×10^1	$< 2 \times 10^{-16}$ ***
bathrooms	1.461×10^4	2.845×10^3	5.134×10^0	2.87×10^{-7} ***
sqft_living	1.471×10^5	4.093×10^3	3.594×10^1	$< 2 \times 10^{-16}$ ***
sqft_lot	5.957×10^3	2.023×10^3	2.945×10^0	3.23×10^{-3} **
floors	3.164×10^3	1.995×10^3	1.586×10^0	0.113
waterfront	6.047×10^5	1.763×10^4	3.430×10^1	$< 2 \times 10^{-16}$ ***
view	4.406×10^4	1.668×10^3	2.641×10^1	$< 2 \times 10^{-16}$ ***
condition	1.827×10^4	1.570×10^3	1.164×10^1	$< 2 \times 10^{-16}$ ***
grade	1.073×10^5	2.572×10^3	4.174×10^1	$< 2 \times 10^{-16}$ ***
sqft_above	1.900×10^4	3.712×10^3	5.119×10^0	3.10×10^{-7} ***
sqft_basement	NA	NA	NA	NA
zipcode	-2.920×10^4	1.798×10^3	-1.624×10^1	$< 2 \times 10^{-16}$ ***
lat	8.648×10^4	1.514×10^3	5.712×10^1	$< 2 \times 10^{-16}$ ***
long	-3.515×10^4	1.878×10^3	-1.871×10^1	$< 2 \times 10^{-16}$ ***
sqft_living15	1.092×10^4	2.408×10^3	4.535×10^0	5.80×10^{-6} ***
sqft_lot15	-1.011×10^4	2.039×10^3	-4.957×10^0	7.23×10^{-7} ***
age	5.985×10^4	2.105×10^3	2.844×10^1	$< 2 \times 10^{-16}$ ***
total_sqft	NA	NA	NA	NA
bath_per_bed	2.120×10^4	1.931×10^3	1.098×10^1	$< 2 \times 10^{-16}$ ***
total_rooms	NA	NA	NA	NA
sqft_diff_15	NA	NA	NA	NA

Table 1: Multivariate Linear Regression Data Summary

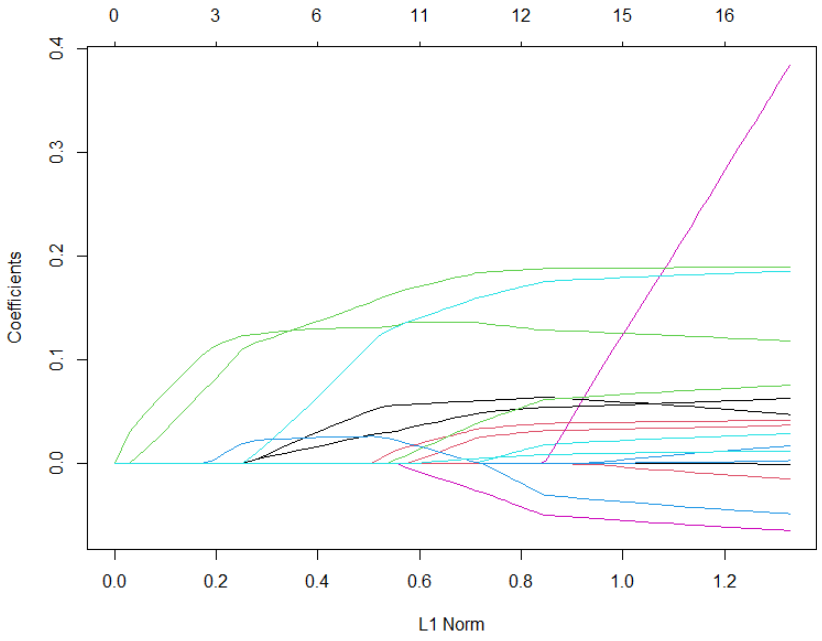

Figure 6: Multivariate Poisson regression coefficient path

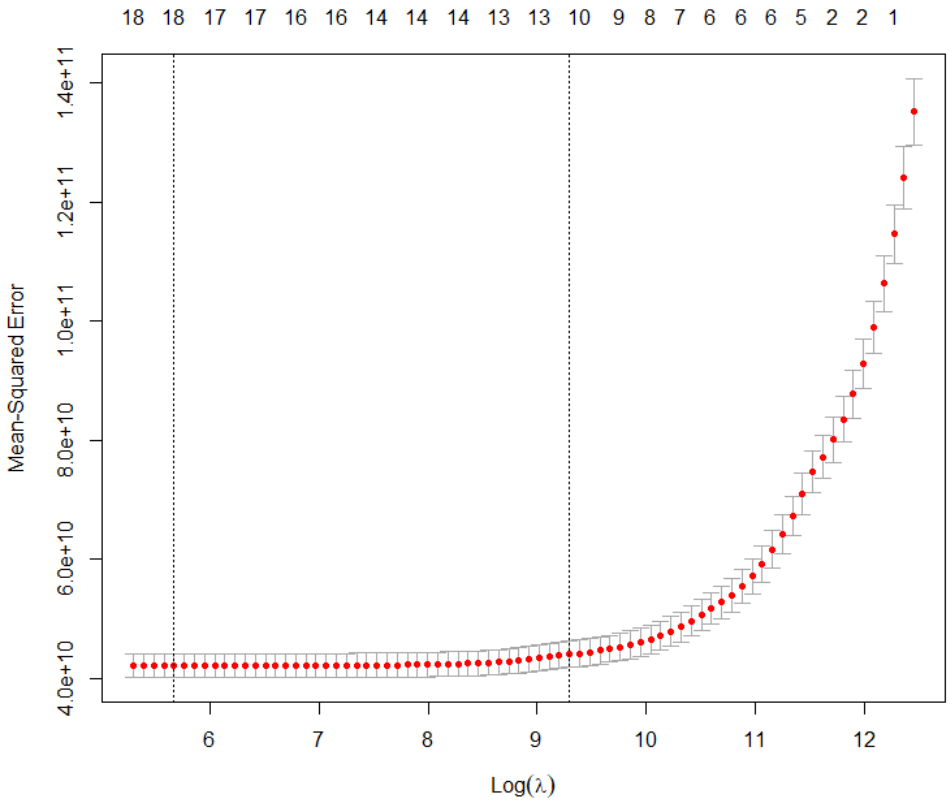

Figure 7: Cross validation plot for Poisson regression

Variable	**Coefficient**
Intercept	536525.639
Bedrooms	-1009.958
Bathrooms	.
Sqft_Living	145781.906
Sqft_Lot	.
Floors	.
Waterfront	526574.268
View	40996.640
Condition	11319.671
Grade	109439.811
Sqft_Above	.
Sqft_Basement	.
Zipcode	.
Lat	74933.499
Long	-11647.660
Sqft_Living15	6652.210
Sqft_Lot15	.
Age	38344.499

Table 2: Coefficients from the Poisson model

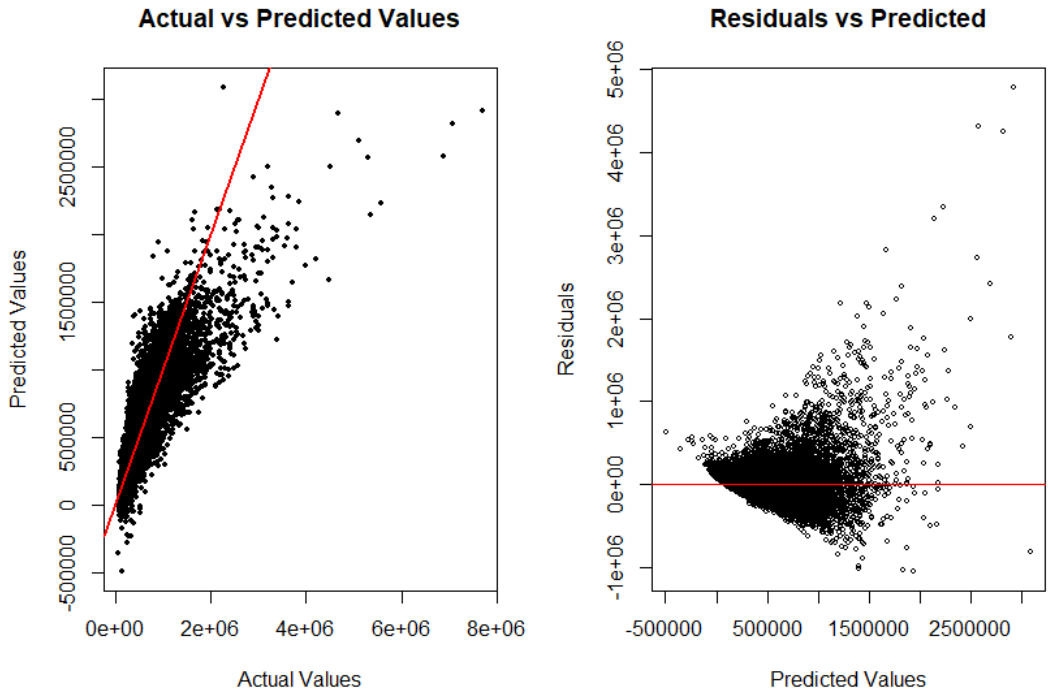

Figure 8: Poisson regression performance

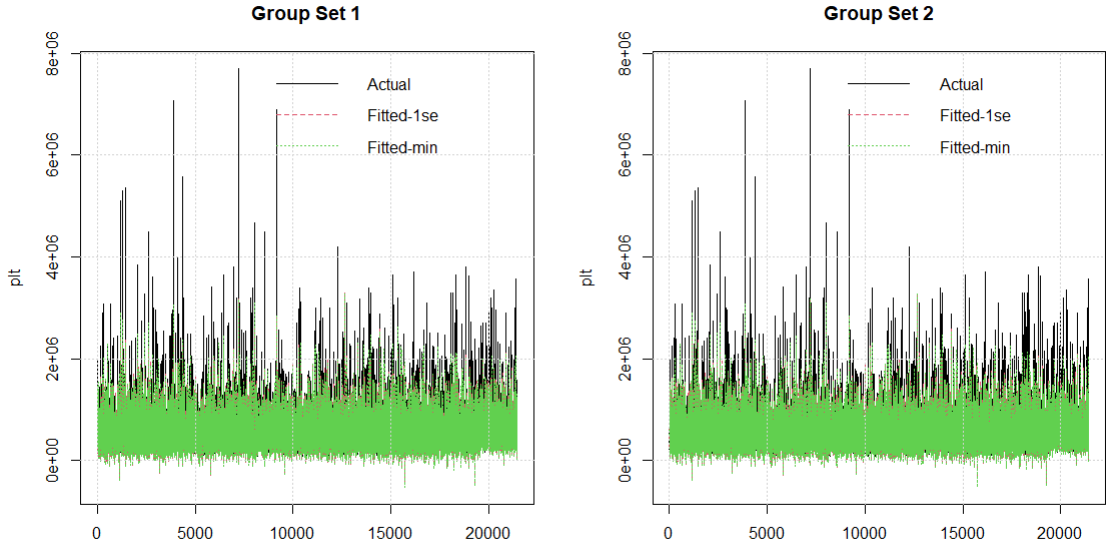

Figure 9: Grouped models predictions

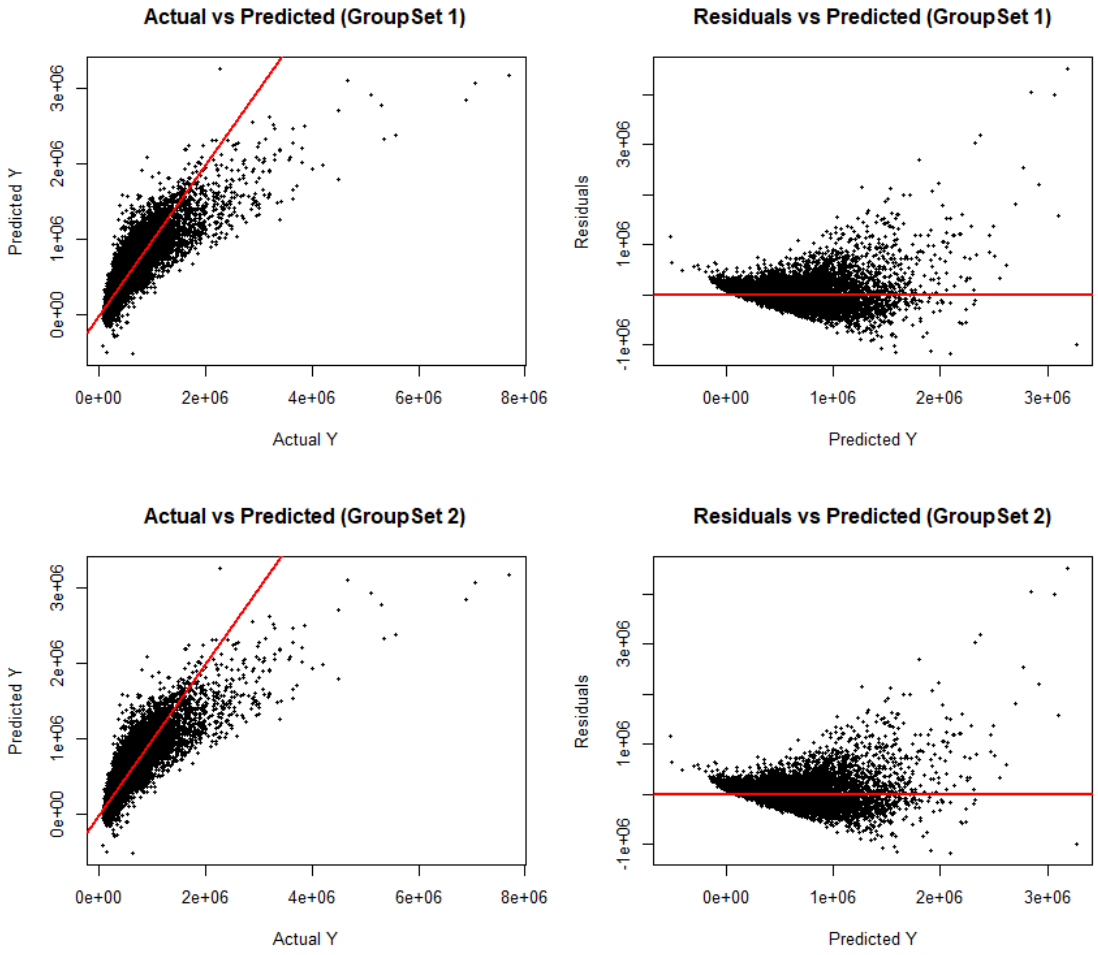

Figure 10: Grouped models performances(only min lambda)

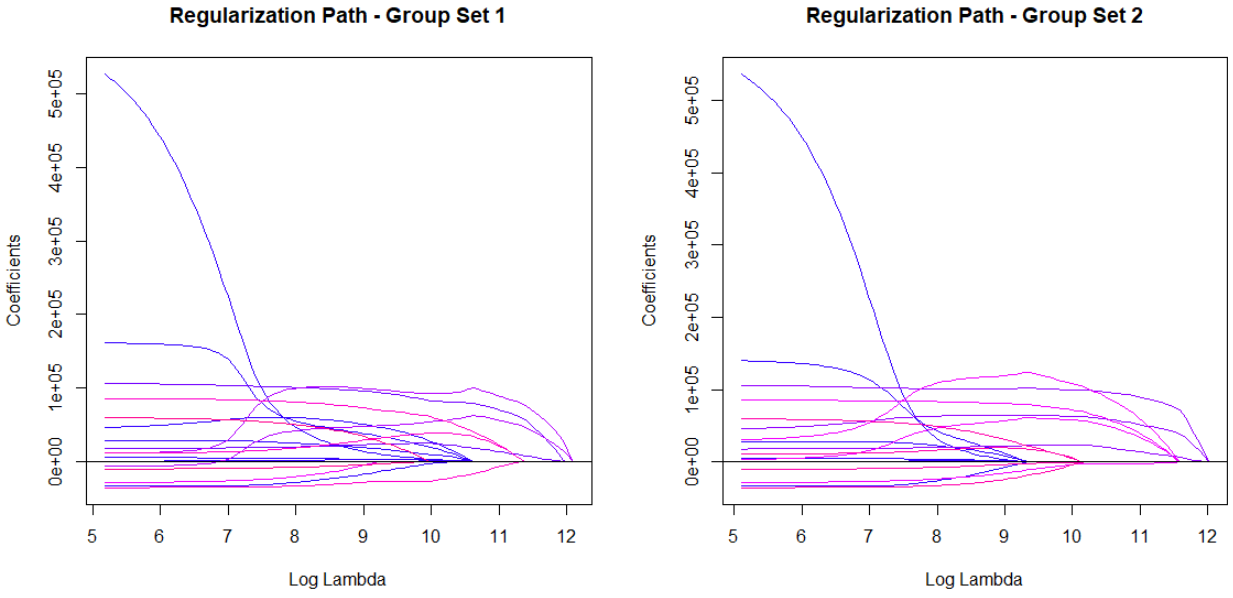

Figure 11: Coefficient path for Group Lasso

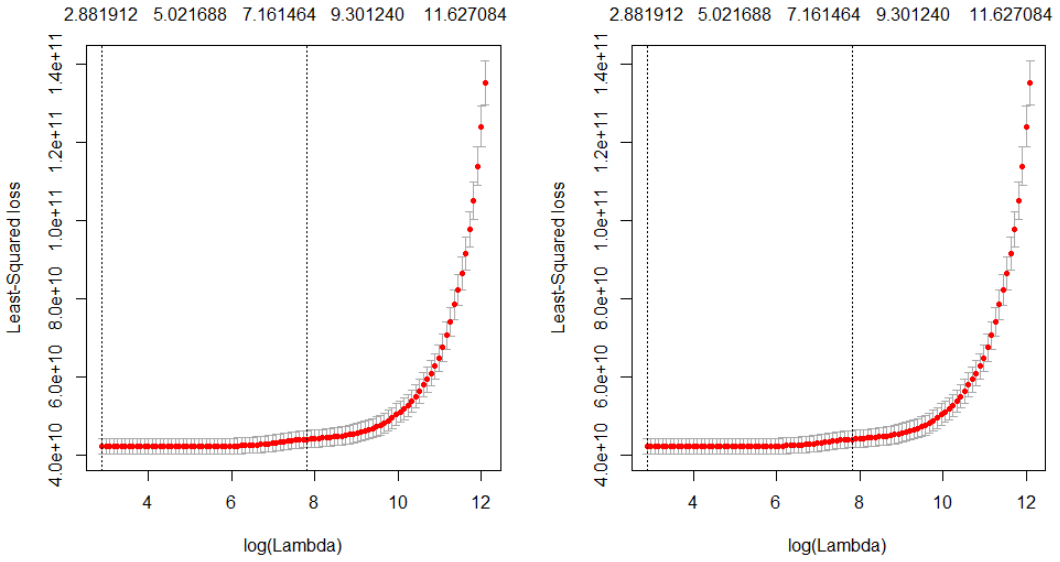

Figure 12: Cross Validation for Group Lasso

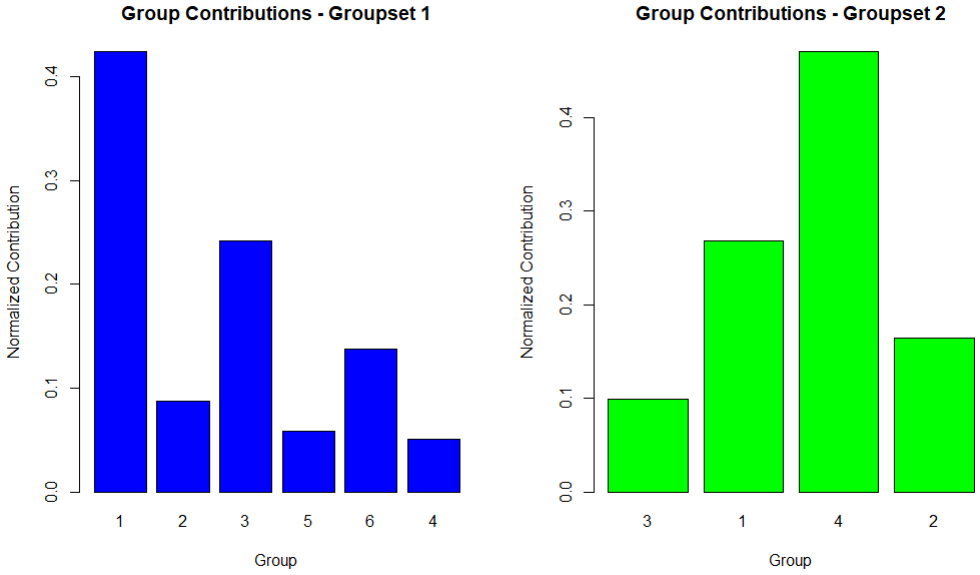

Figure 13: Group contribution for gglasso models

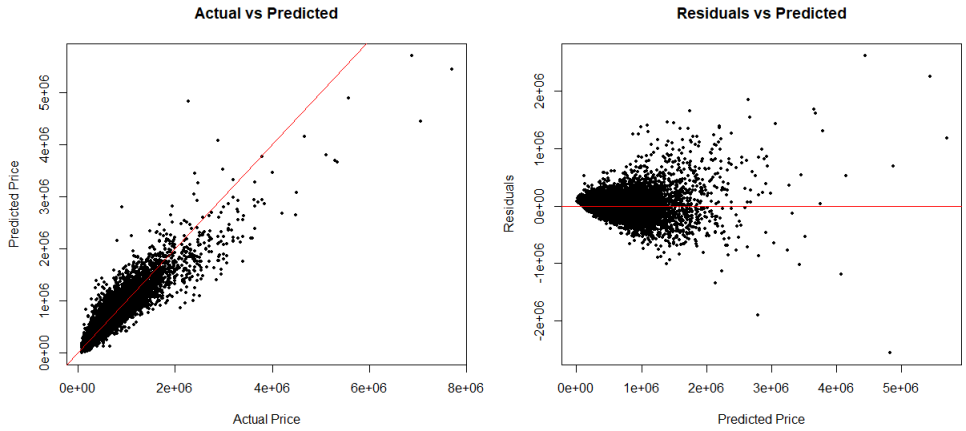

Figure 14: GAM performance

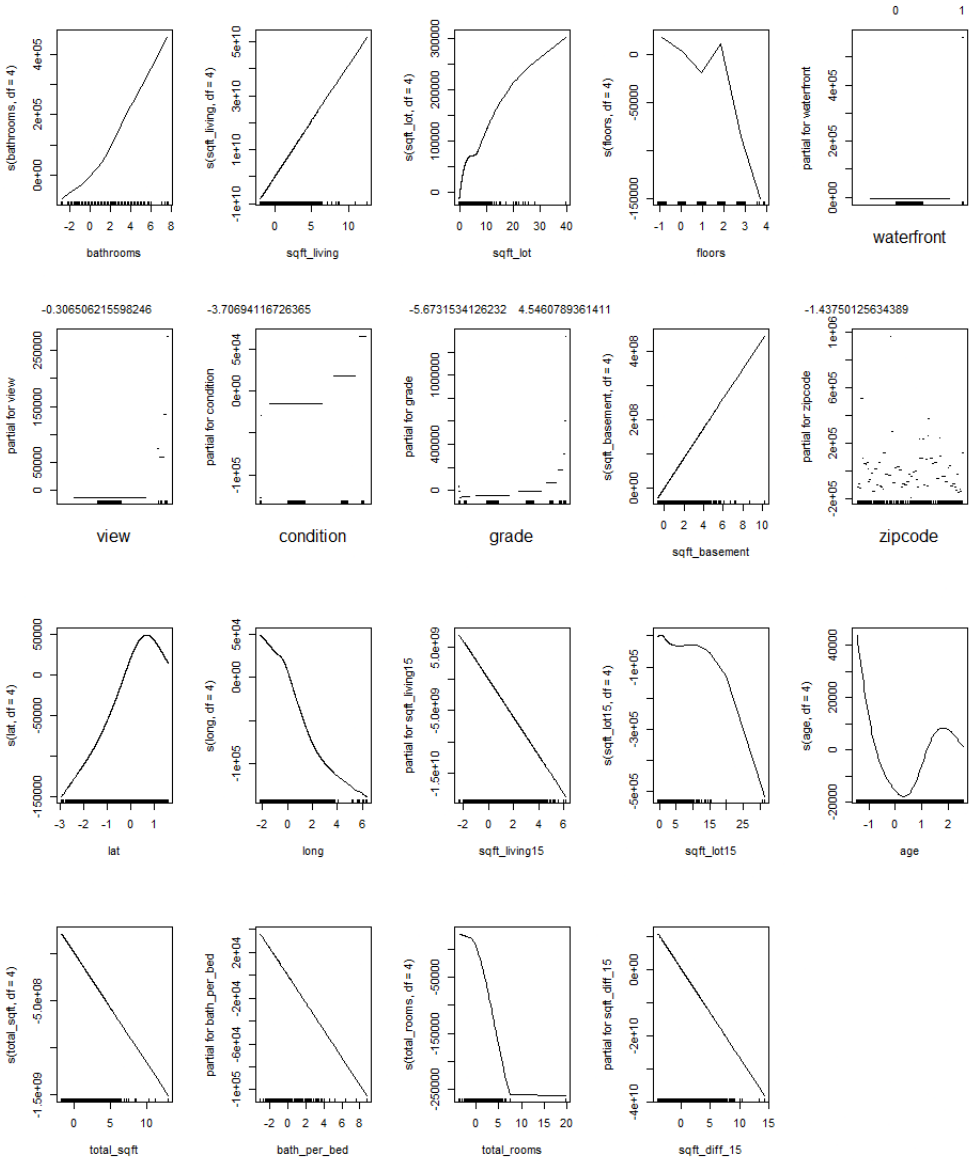

Figure 15: GAM/splines: price vs each feature

Variable	**Df**	**Sum Sq**	**Mean Sq**	**F value**	**P(>F)**
s(bathrooms, df = 4)	1	8.03×10^{14}	8.03×10^{14}	39053.23	$< 2.2 \times 10^{-16}$ ***
s(sqft_living, df = 4)	1	5.96×10^{14}	5.96×10^{14}	28967.84	$< 2.2 \times 10^{-16}$ ***
s(sqft_lot, df = 4)	1	1.51×10^{12}	1.51×10^{12}	73.61	$< 2.2 \times 10^{-16}$ ***
s(floors, df = 4)	1	1.36×10^{11}	1.36×10^{11}	6.61	0.0102 *
waterfront	1	1.02×10^{14}	1.02×10^{14}	4962.16	$< 2.2 \times 10^{-16}$ ***
view	4	5.78×10^{13}	1.45×10^{13}	702.67	$< 2.2 \times 10^{-16}$ ***
condition	4	1.58×10^{13}	3.96×10^{12}	192.59	$< 2.2 \times 10^{-16}$ ***
grade	11	1.89×10^{14}	1.72×10^{13}	836.48	$< 2.2 \times 10^{-16}$ ***
s(sqft_basement, df = 4)	1	3.98×10^{12}	3.98×10^{12}	193.72	$< 2.2 \times 10^{-16}$ ***
zipcode	69	4.74×10^{14}	6.86×10^{12}	333.79	$< 2.2 \times 10^{-16}$ ***
s(lat, df = 4)	1	8.43×10^{11}	8.43×10^{11}	40.98	1.57×10^{-10} ***
s(long, df = 4)	1	7.32×10^{11}	7.32×10^{11}	35.58	2.49×10^{-9} ***
sqft_living15	1	1.65×10^{12}	1.65×10^{12}	80.44	$< 2.2 \times 10^{-16}$ ***
s(sqft_lot15, df = 4)	1	2.35×10^{11}	2.35×10^{11}	11.43	0.0007 ***
s(age, df = 4)	1	4.43×10^{11}	4.43×10^{11}	21.53	3.50×10^{-6} ***
s(total_sqft, df = 4)	1	3.09×10^{11}	3.09×10^{11}	15.02	1.06×10^{-4} ***
bath_per_bed	1	2.05×10^{9}	2.05×10^{9}	0.10	0.752
s(total_rooms, df = 4)	1	5.97×10^{11}	5.97×10^{11}	29.04	7.17×10^{-8} ***
sqft_diff_15	1	5.42×10^{11}	5.42×10^{11}	26.33	2.90×10^{-7} ***
Residuals	21299	4.38×10^{14}	2.06×10^{10}		

Table 3: GAM: ANOVA for Parametric Effects

Variable	**Npar Df**	**Npar F**	**P(>F)**
s(bathrooms, df = 4)	3	22.99	7.44×10^{-15} ***
s(sqft_living, df = 4)	3	76.91	$< 2.2 \times 10^{-16}$ ***
s(sqft_lot, df = 4)	3	21.60	5.84×10^{-14} ***
s(floors, df = 4)	3	24.35	9.99×10^{-16} ***
s(sqft_basement, df = 4)	3	70.34	$< 2.2 \times 10^{-16}$ ***
s(lat, df = 4)	3	51.48	$< 2.2 \times 10^{-16}$ ***
s(long, df = 4)	3	15.16	7.52×10^{-10} ***
s(sqft_lot15, df = 4)	3	14.56	1.82×10^{-9} ***
s(age, df = 4)	3	61.05	$< 2.2 \times 10^{-16}$ ***
s(total_sqft, df = 4)	3	333.48	$< 2.2 \times 10^{-16}$ ***
s(total_rooms, df = 4)	3	24.54	7.77×10^{-16} ***

Table 4: GAM: ANOVA for Nonparametric Effects

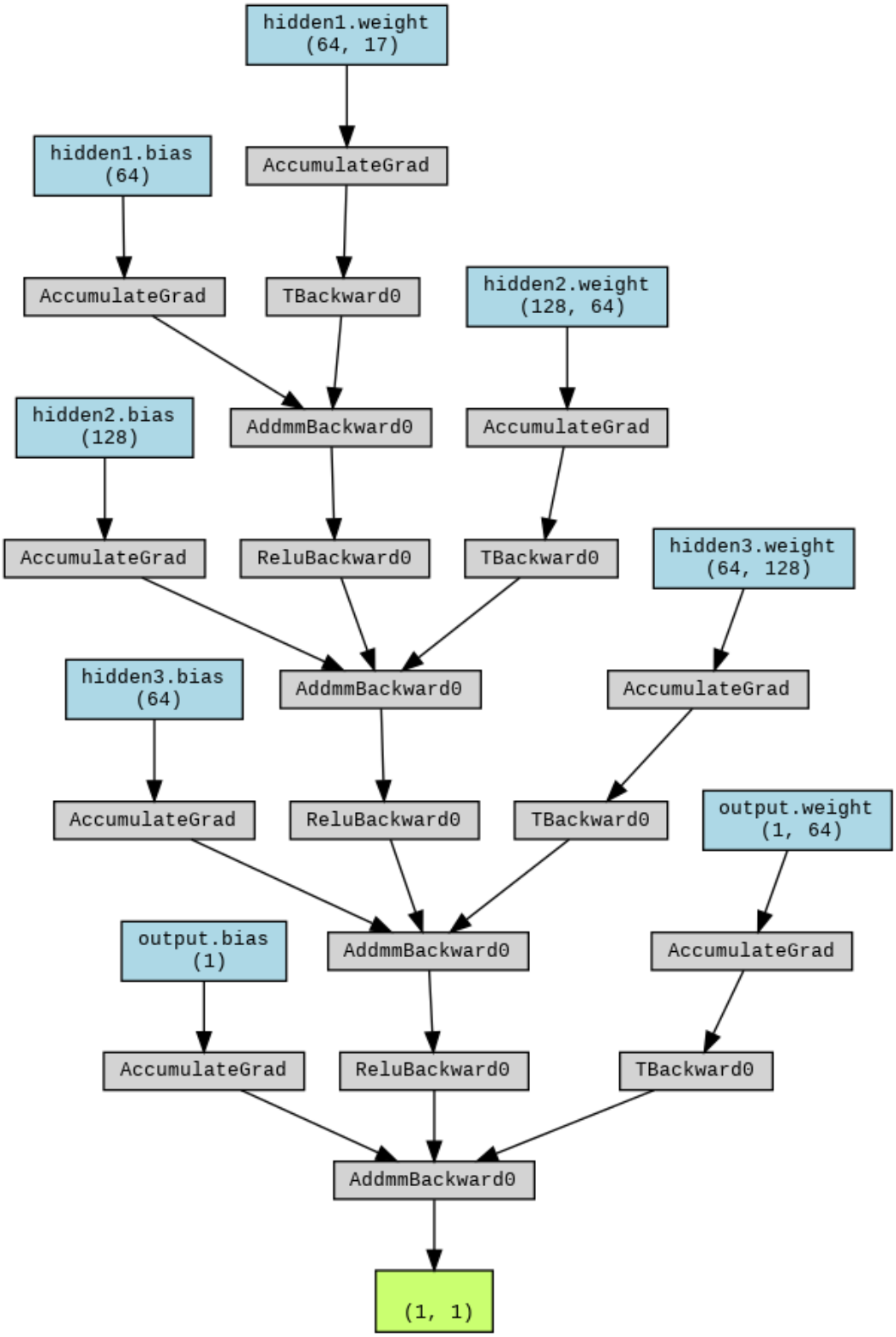

Figure 16: Neural Network architecture

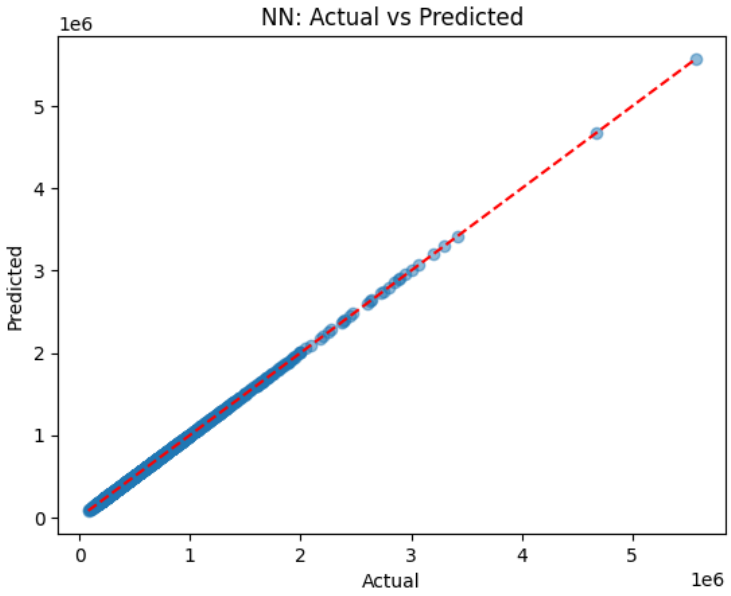

Figure 17: Neural Network actual vs predicted

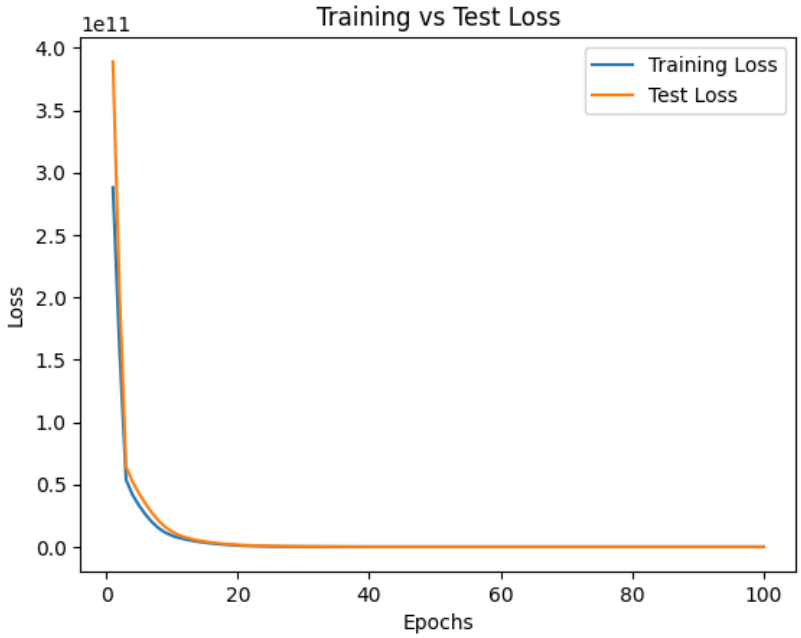

Figure 18: Neural Network training