Boston Housing dataset

Pritom

12/13/2021

Dataset

This data set is from **Kaggle**. Basic description of this data set is, it contains total 79 features. Fror which modeling become a cumbersome task.

Missing values

variables	value
pool_qc	99.5
misc_feature	96.2
alley	93.7
fence	80.1
fireplace_qu	47.8
lot_frontage	18.1
garage_type	5.9
garage_yr_blt	5.9
garage_finish	5.9
garage_qual	5.9

So, we will eliminate any of the variables for which the missing value percentage is more than 20%. So we are going to eliminate variables

```
- PoolQC
- MiscFeature
- Alley
- Fence
- FireplaceQu
```

Selecting only the variables with completation rate more than 80%

Since it is very risky to retrieve the data when there is more than 20% observations are missing. So we simply ignore the columns.

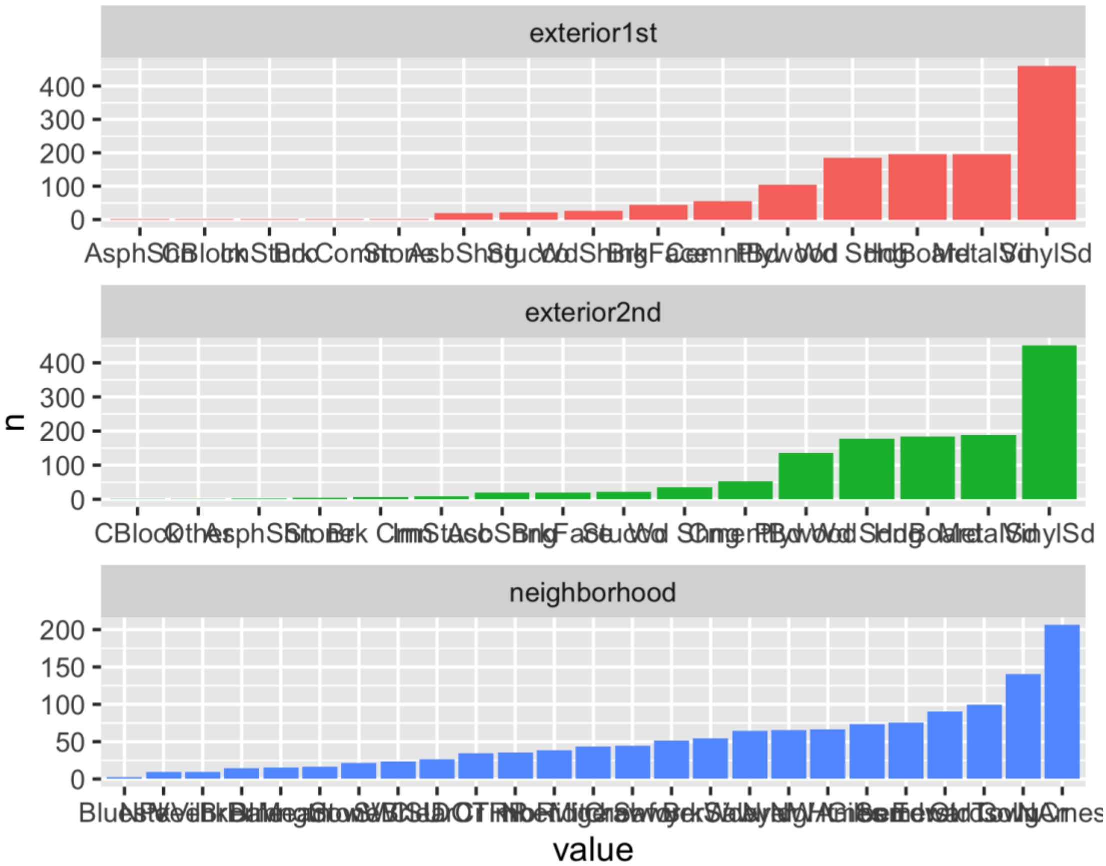
Categorical variables

There are 38 total categorical variables. Among which some of them have more than 10 levels. Which may be redundant for the ML models. SO we simply lump the factors which has unnecessary labels.

```
## # A tibble: 38 × 2
##    name           `Number of caterories`
##    <chr>                           <int>
##  1 neighborhood                       25
##  2 exterior2nd                        16
##  3 exterior1st                        15
##  4 condition1                          9
##  5 sale_type                           9
##  6 condition2                          8
##  7 house_style                         8
##  8 roof_matl                           8
##  9 bsmt_fin_type1                      7
## 10 bsmt_fin_type2                      7
## # … with 28 more rows
```

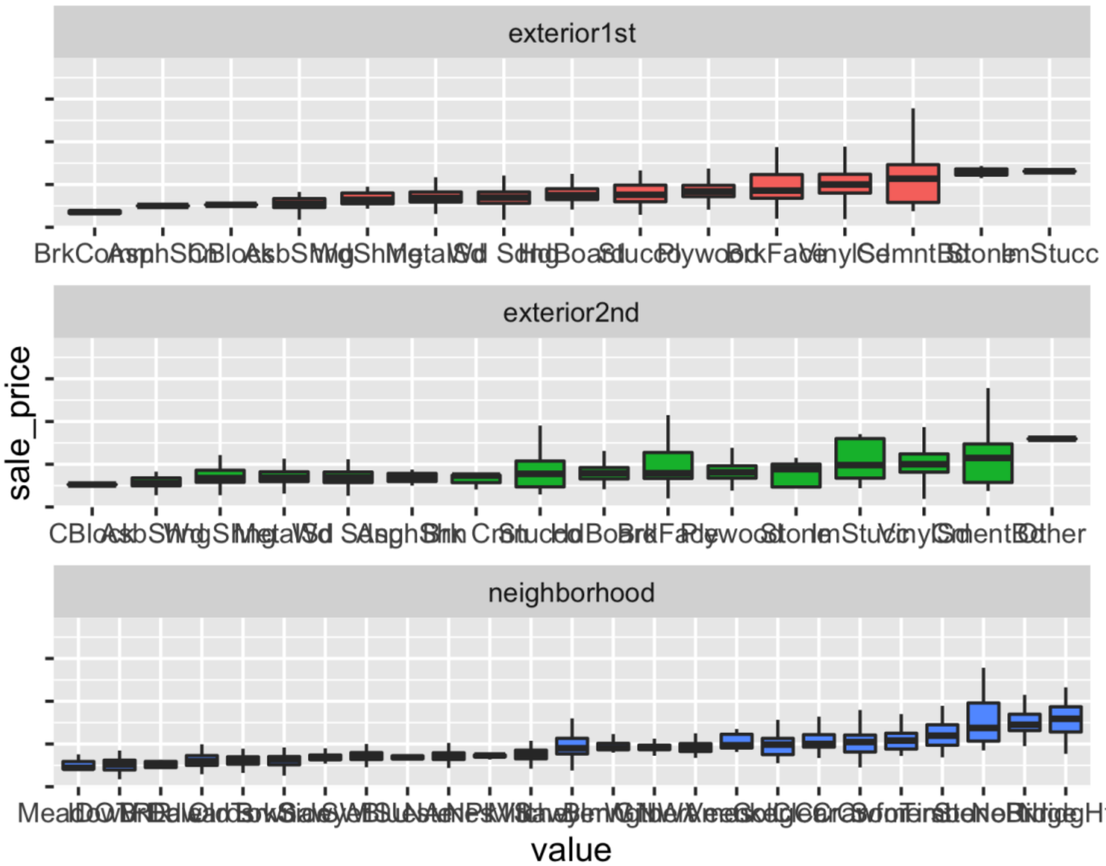
So variable neighborhood, exterior2nd and exterior1st has extra amount of variables. So we will manually check whether all those laves could be categorise as lower number of categories.

Barplot for showing the distributions



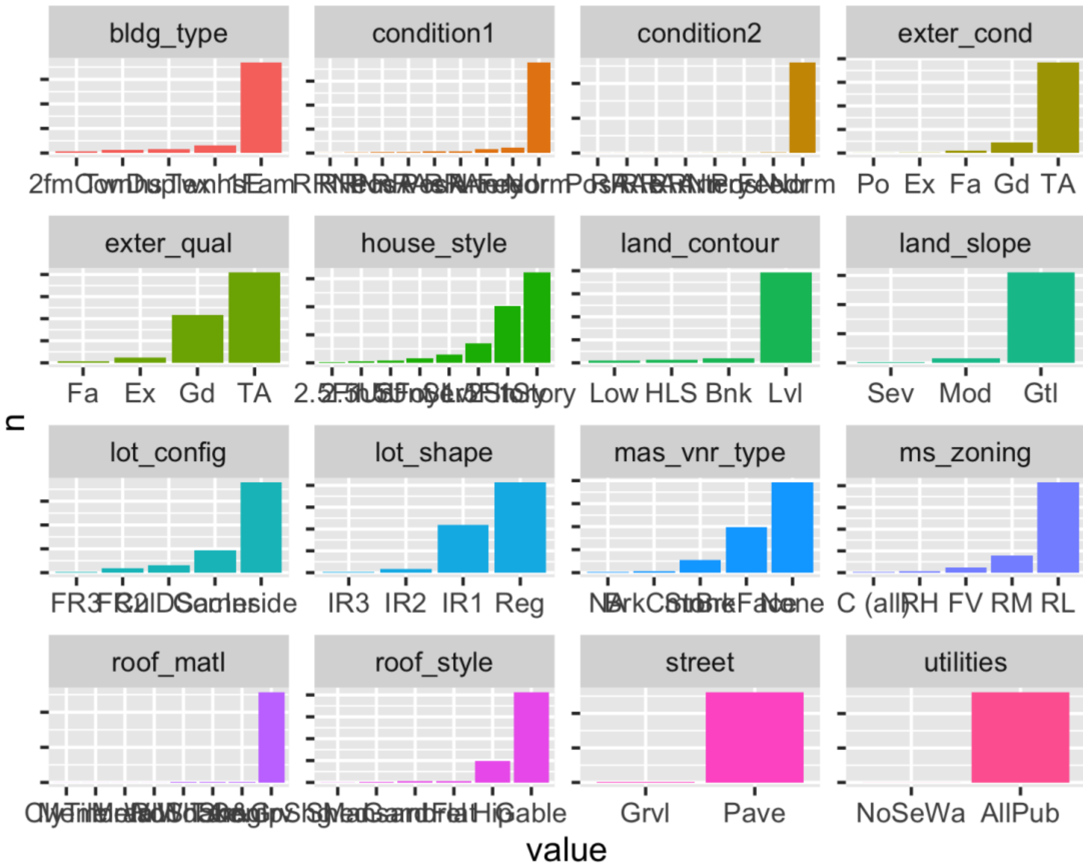
This Barplot shows that there are many labels that have a significantly lower count and which may introduce “Zero Variance” problem to the variables when we will apply dummy encoding. SO we need to lump those labels together with lower count.

Boxplot for the effects of each labels.



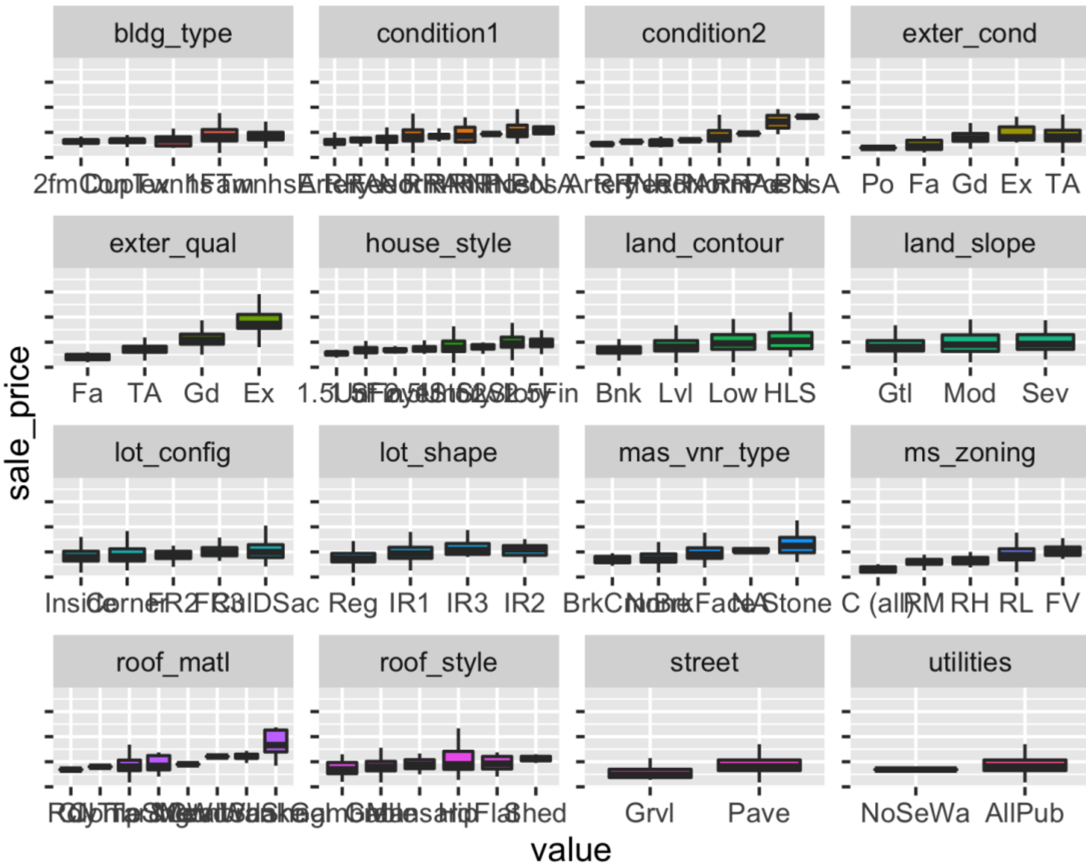
We can see that there exist different mean levels for those varoables above. SO those variables may be useful for the prediction.

Barplot for other categorical variables



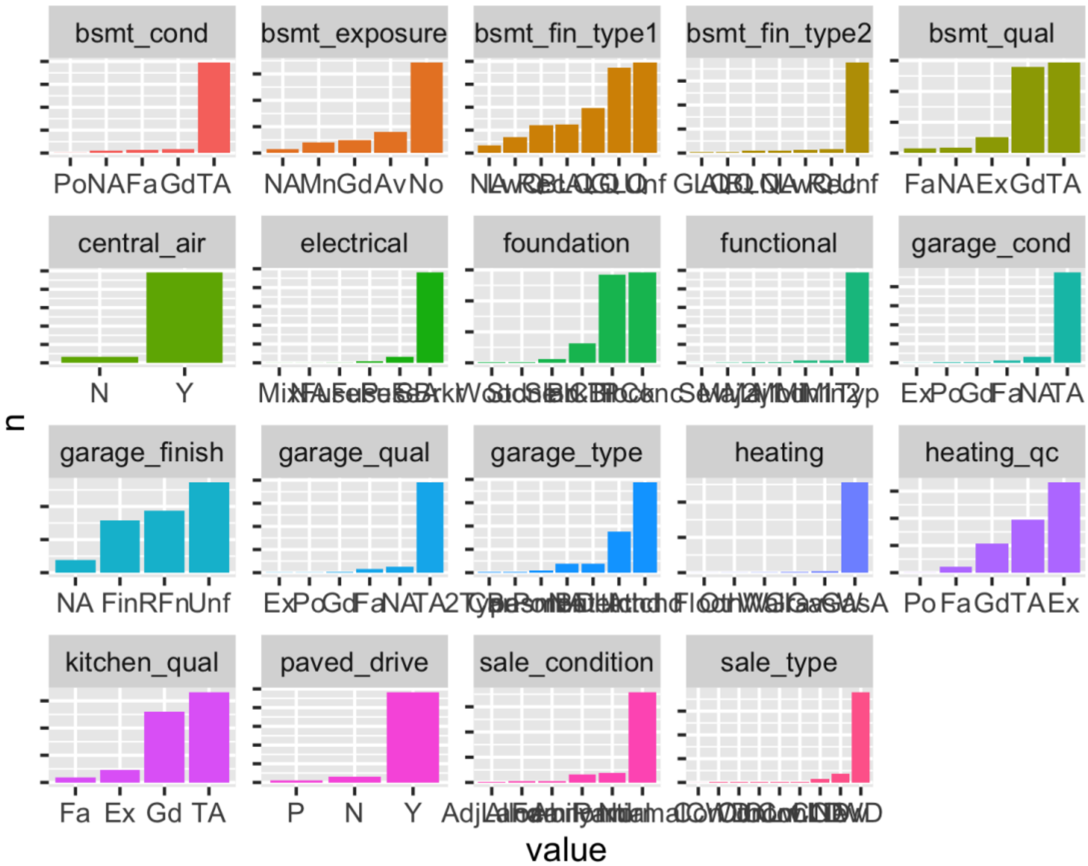
Again, we can see that there are different number of counts for the different labels of the variables.

Boxplot for those categorical variables



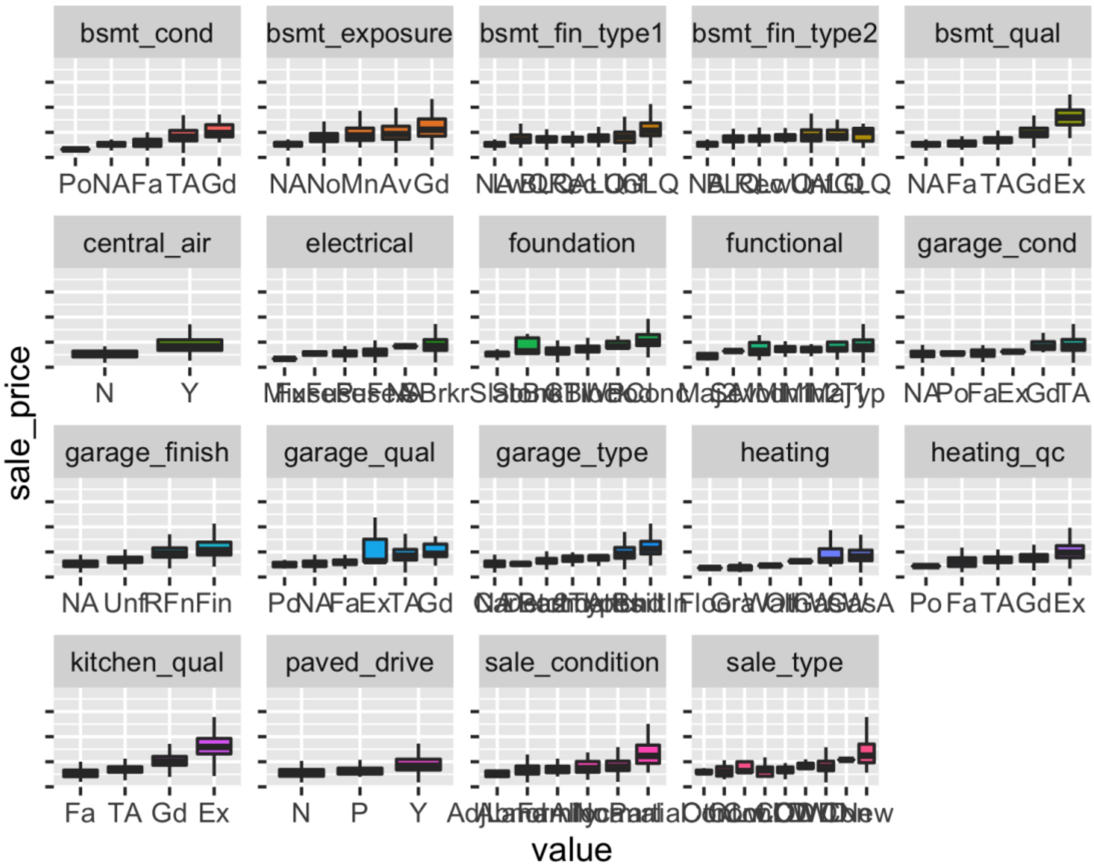
And here we can see that many of those varoables can be helpful for the predictive performance. Because there are different mean levels for the different labels for some variables.

Barplot for other categorical variables



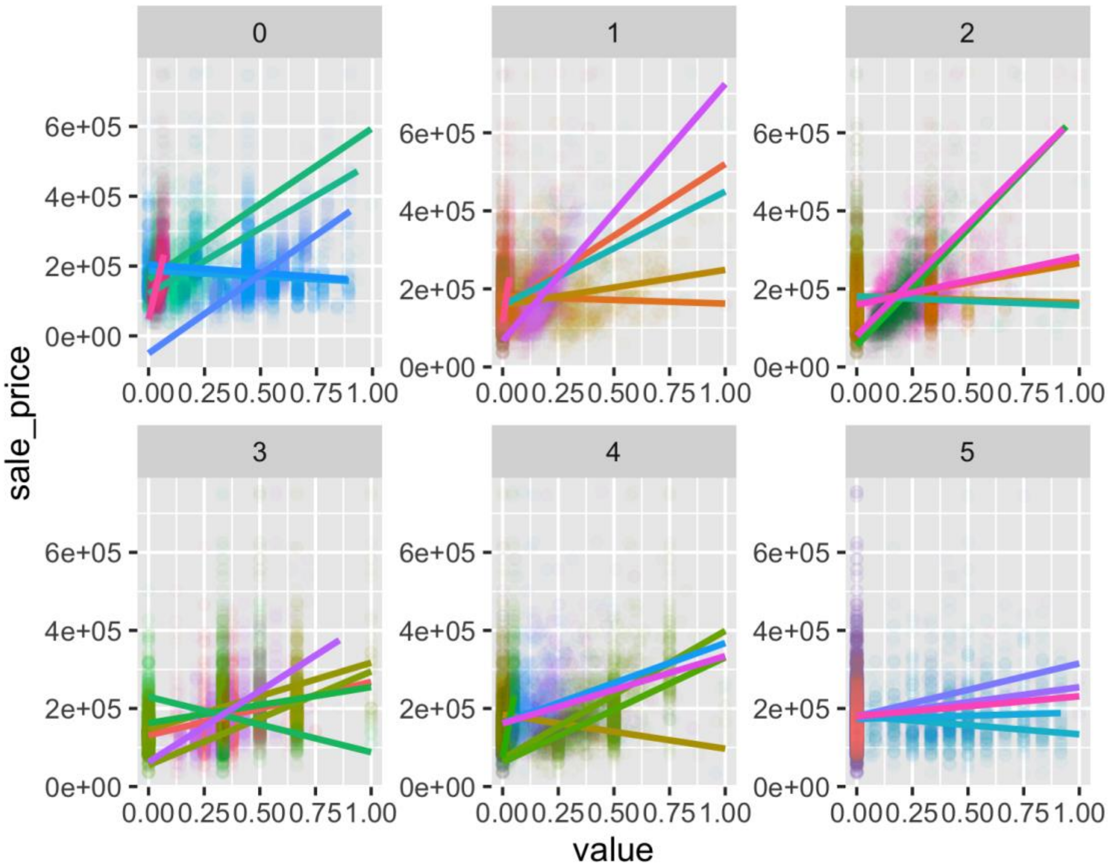
we can see that there are different amount of counts for the different labels of the variables.

Boxplot for other categorical variables



Here we can see that many of those varoables can be helpful for the predictive performance. Because there are different mean levels for the different labels for some variables.

Scatter plot for the numeric variables



Here we can see that different numerical variable showes a strong linear association with the predictor variables.

Correlation among the numeric variables

```
## # A tibble: 10 × 3
##    rowname        name           value
##    <chr>          <chr>          <dbl>
##  1 garage_cars    garage_area     0.88
##  2 gr_liv_area    tot_rms_abv_grd 0.82
##  3 total_bsmt_sf  x1st_flr_sf     0.81
##  4 overall_qual   sale_price      0.79
##  5 gr_liv_area    sale_price      0.71
##  6 x2nd_flr_sf    gr_liv_area     0.69
##  7 bedroom_abv_gr tot_rms_abv_grd 0.68
##  8 bsmt_fin_sf1   bsmt_full_bath  0.65
##  9 garage_cars    sale_price      0.64
## 10 garage_area    sale_price      0.63
```

There are many correlated terms. So we might eliminate some of the correlated variables.

Modeling

Defining the models

We will use tidymodel framework in R. This is the most recent TidyWorkflow by the Rstudio community. To use that framework we need to define the models first.

Cross validation and recipe

To make data pre-processing easy, we will use the recipe package and to train the hyperparameters we will use the rsample package. We will use 6 fold cross validation for this.

Function for model fitting

Defining a function for the modelling, which will automatically find out the best hyperparameters settings and fit the model.

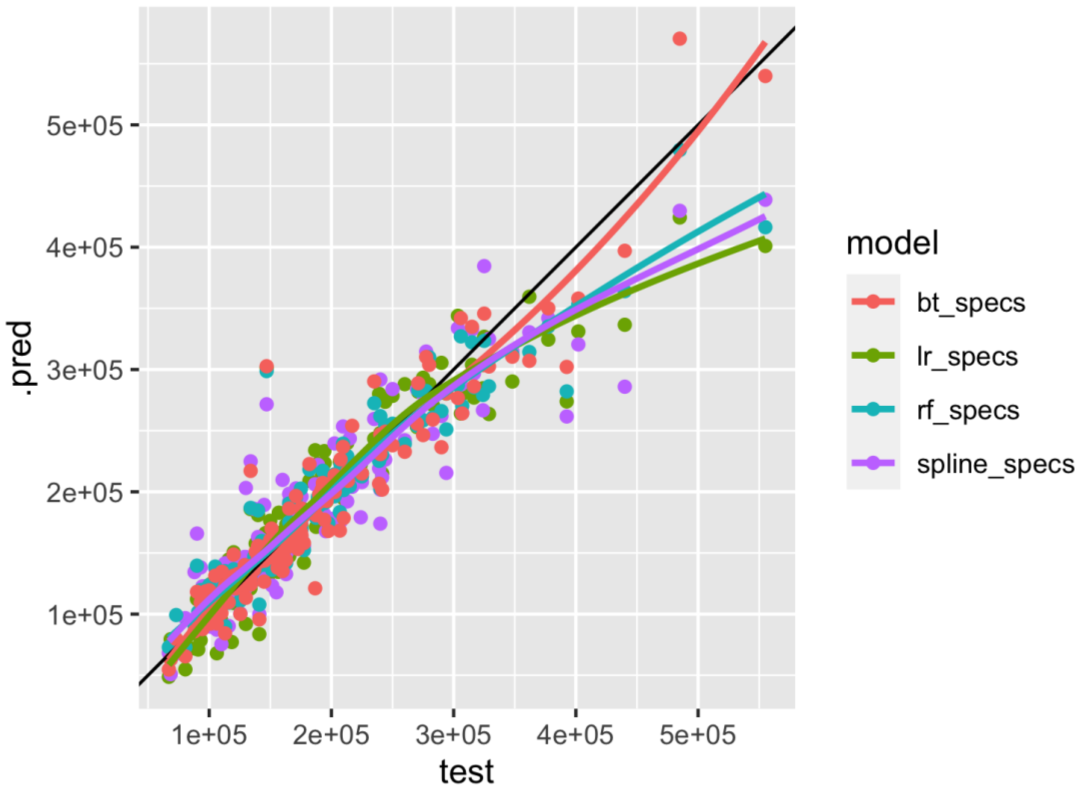
Model fitting

We will fit the model and save that file in an Rds version so that we dont need to run those model again and again.

Predicting the test dataset

We already train those model on the basis of the train dataset. Now we will evaluate the performance on the test dataset and find out which model is performing best for this dataset,

Predicted value vs estimated values



This graph can be explained as the closer the point lie on the diaonal line the better the prediction is. We can see that there is a tendency to underestimate the price of the house where the price is bit higher. SO none of those models are performing well. We may need to try some other approach.

Mean absolute error

```
## # A tibble: 4 × 4
##   model        .metric .estimator .estimate
##   <chr>        <chr>   <chr>          <dbl>
## 1 rf_specs     mae     standard      16356.
## 2 bt_specs     mae     standard      17252.
## 3 lr_specs     mae     standard      20551.
## 4 spline_specs mae     standard      24108.
```

One of the metric for the evaluation of the regression prediction is “MAE”. On that index the models svm, logistic regression and random forest provide almost equal performance. So we will choose any of those 3 as our final model.

Root mean square error

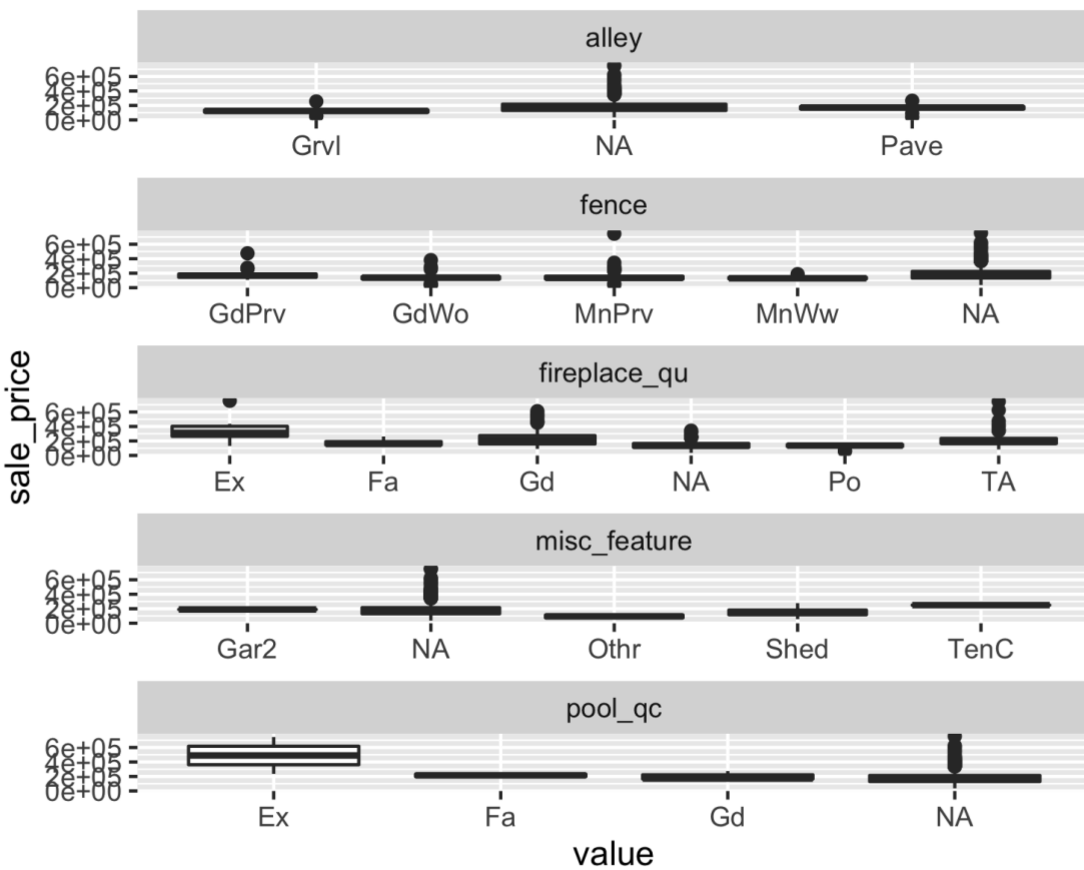
```
## # A tibble: 4 × 4
##   model        .metric .estimator .estimate
##   <chr>        <chr>   <chr>          <dbl>
## 1 bt_specs     rmse    standard      26564.
## 2 rf_specs     rmse    standard      27239.
## 3 lr_specs     rmse    standard      31436.
## 4 spline_specs rmse    standard      35085.
```

RMSE is an another matric for the exaluation of the regression performance. From here we can choose Ranndomforest for our desirable model since it has the highest accuracy.

Working with the kaggle test data

This dataset is provided by kaggle for the submission purpose to see the overall ranking compare to the other users world wide. So my estimate from this model gives me a rank of 4000+

Impatcs of Missing values



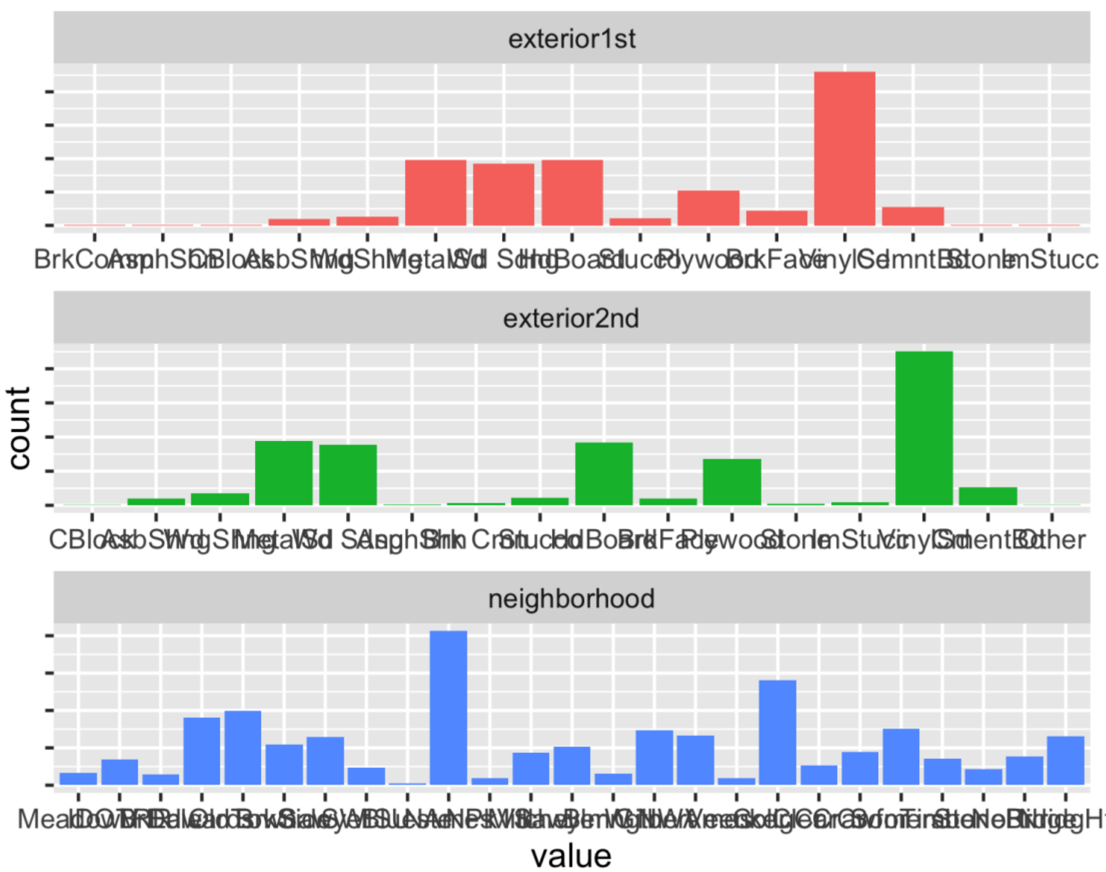
From this plot we can see that there may be some impact of the missing value. Since the mean level of the missing values differ from other labels for some variables. So we will follow the same steps only consider the missing values with different categories.

Trying a different approach

This time we will try to find whether treating the missing values as a different category help to increase the performance or not.

Storing the original data

Motivation for new approach



At the first look this graph may seems messy. But this can me interpreted like at the far left there the levels with the least median level of sales price. So we will perform the lumping process so that the similar categories that have a queal median level lump together..

Defining the function for lumping

```
## # A tibble: 16 × 2
##    exterior2nd exterior2nd_edt
##    <chr>       <fct>
##  1 CBlock      frac1
##  2 AsbShng     frac1
##  3 Wd Shng     frac1
##  4 MetalSd     frac2
##  5 Wd Sdng     frac2
##  6 AsphShn     frac2
##  7 Brk Cmn     frac2
##  8 Stucco      frac3
##  9 HdBoard     frac3
## 10 BrkFace     frac3
## 11 Plywood     frac4
## 12 Stone       frac4
## 13 ImStucc     frac4
## 14 VinylSd     frac6
## 15 CmentBd     frac6
## 16 Other       frac6
```

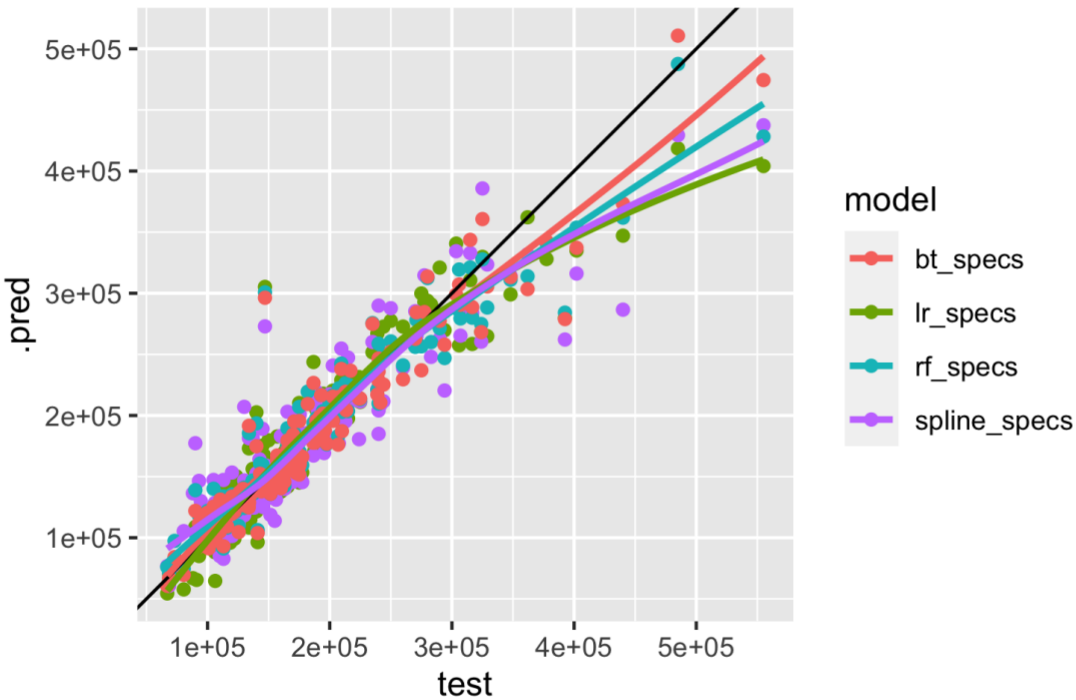
Changing the levels of each variables.

Cross validation and recipe

Model fitting

```
## [1] "hyperParameter Training"
## [1] "Model Training"
## [1] "hyperParameter Training"
## [1] "Model Training"
## [1] "hyperParameter Training"
## [1] "Model Training"
## [1] "hyperParameter Training"
## [1] "Model Training"
```

Predicting the test dataset



We can see that there is a tendency to underestimate the price of the house where the price is bit higher.

MAE

```
## # A tibble: 4 × 4
##   model        .metric .estimator .estimate
##   <chr>        <chr>   <chr>          <dbl>
## 1 bt_specs     mae     standard      16074.
## 2 rf_specs     mae     standard      16218.
## 3 lr_specs     mae     standard      20717.
## 4 spline_specs mae     standard      24398.
```

So the models xgBoost provide almost good performance. So we will choose xgBoost as our final model.

RMSE

```
## # A tibble: 4 × 4
##   model        .metric .estimator .estimate
##   <chr>        <chr>   <chr>          <dbl>
## 1 bt_specs     rmse    standard      25537.
## 2 rf_specs     rmse    standard      26876.
## 3 lr_specs     rmse    standard      31264.
## 4 spline_specs rmse    standard      34971.
```

Here we can see some anomaly. In this metric Random forest however perform better that the boosted trees. But some how the previous metric looks more convenient. So we will go with the xgboost model.

Working with the kaggle test data

My final estimate from this model gives me a rank of 2142. Which is a great jump. But still there are much room for the improvement. Carefully feature extraction will provibe more better prediction performance.