**House Price Prediction**

**Date:** 04/25/2023

**Team:**

Meghana Bathula (U00866246)

Shiva Ram Kapil Pabba (U00868229)

Venkat Narsimha Reddy Velga (U00864830)

**Title**: House price prediction in king county, USA.

**Objective:** Most of the population across the globe have plans on buying a house one day, and they might not be sure about the price they can invest. To address such problems, we want to build a model that could predict the price. As we don’t have the global data, for now our predictions are limited to King County, USA. The predictions are based on the model that we constructed and to train the model we have used the below dataset from Kaggle <https://www.kaggle.com/datasets/harlfoxem/housesalesprediction?resource=download>

**Data:**

This dataset has 18 input variables.

1. Number of Bedrooms
2. Number of Bathrooms
3. Square footage of the living area
4. Square footage of the lot
5. Number of Floors
6. Waterfront
7. View,
8. Condition
9. Grade
10. Square footage above
11. Square footage of the basement
12. Year the house was built
13. Renovation Year
14. Zip code
15. Latitude
16. Longitude
17. Average sq ft of living area of closest 15 houses
18. Average sq ft of the lot of closest 15 houses

And the label is “Price” and there are 21,613 observations for this dataset.

To check if there was any missing data, we used missmap function from *Amelia* library and could see that there is no missing data in the dataset as shown below.

A picture containing chart

Description automatically generated

Let us also visualize the data using a correlation plot where price is the outcome and remaining variables will be the input features.

Text

Description automatically generated

Using the input variables, we want to predict the price, to do that we tried using different models to see which fits best on this data, they are as below.

1. Generalized additive model
2. Bagging
3. Boosting
4. Principle component analysis

Let us explore each model against the dataset.

1. **Generalized additive model**: First let us try to visualize the data using a scatter plot of all the independent variables vs Price. In this the blue line represents the linear regression function fitted to the data. As we can see the linear regression couldn’t fit the entire data. The variables condition, floors, view, waterfront and yr\_renovated have less than or equal to 5 values. So smoothing splines are not used on them.

Graphical user interface, text

Description automatically generated

**A picture containing diagram

Description automatically generated**

By fitting the basic linear regression model using all variables, we get Adjusted R squared values as 0.6938.

Text, table

Description automatically generated

Fitting the GAM model to the dataset variables individually based on the type of variable.

Text

Description automatically generated

Text

Description automatically generated with medium confidence

Adjusted R squared values has increased slightly compared to linear regression model when used natural splines.

We are now applying smoothing splines to dataset to check whether they fit the model better and produce a better R squared adjusted value.

**Text

Description automatically generated**

**Table

Description automatically generated**

For GAM model using smoothing splines, we get the adjusted R squared value as 0.822 which is higher than the linear regression and the GAM using natural splines.

**Text

Description automatically generated**

The data as you see in the below graph has less residuals for most of the data while some of the data is scattered with higher residuals.

**Chart, scatter chart

Description automatically generated**

1. **Bagging:**  When it comes to bagging (bootstrap aggregating) approach, we created the model using randomforest() method with all the input features on the training data set and with maximum number of terminal nodes in each decision tree as 8 as shown below.

**A picture containing shape

Description automatically generated**

Below is the output of the plot for the out-of-bag error rate as a function of number of trees.

Chart, line chart

Description automatically generated

From above we can see that as the number of trees grows the error rate decreases up to 100 trees and then almost remains constant.

Below is the scatter plot of the actual vs the predicted values of the price using bagging model with a diagonal line, and we can see that there are some points which are far from the diagonal line, and also looking at the summary below, we can say that approximately 65% of the variance is explained by this model.

Text

Description automatically generated

Chart, scatter chart

Description automatically generated

Text

Description automatically generated

1. **Boosting:** When it comes to boosting in order to create the model, we need to define the input parameters like the shrinkage, interaction depth and minimum number of observations at the terminal node, instead of defining the random values for these parameters we wanted iterate through combination of these parameters to find which one yield lowest MSE. To do this we defined a grid that has different combinations of these parameters as below.

**Text

Description automatically generated**

And the grid looks like below.

Table

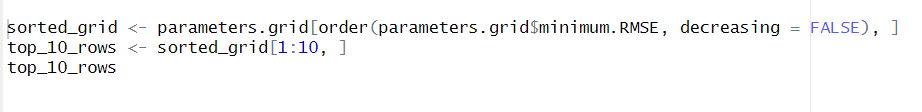
Description automatically generated with low confidence

Now for each row of this grid we try to create a boosting model and get the error and store it in corresponding row.

Text, application

Description automatically generated

Now we sort this grid based on the increasing order of the error rate and see for what parameters we get low error rate.



Table

Description automatically generated with medium confidence

From the above we can see that the error rate is low when the shrinkage is 0.1, interaction depth is 6, and the minimum observations at the terminal node is 15.

Cycle2: Using these values, we try to create another grid and iterate through each row and create a boosting model and get error rate for each row for the second cycle.

Text

Description automatically generated

The grid looks like below.

Table

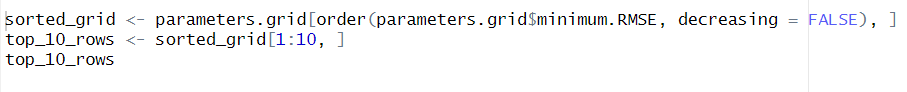
Description automatically generated with medium confidence

Now for each row of this grid we try to create a boosting model and get the error and store it in corresponding row.

**Text, application

Description automatically generated**

Now we sort this grid based on the increasing order of the error rate and see for what parameters we get low error rate.

****

**Table

Description automatically generated**

From the above we can see that the error rate is low when the shrinkage is 0.05, interaction depth is 8, and the minimum observations at the terminal node is 15.

Using the above parameters let’s construct a final boosting model.

Text

Description automatically generated

Here is the summary of the model.

Text, table

Description automatically generated

A picture containing chart

Description automatically generated

Let us predict the price on the test set and plot a scatter plot for the actual vs predicted prices.

Text

Description automatically generated with medium confidence

Chart, scatter chart

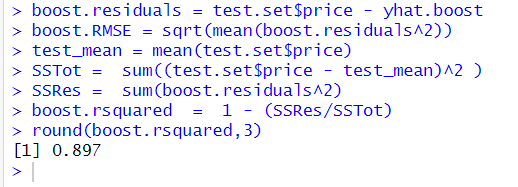
Description automatically generated

Looking at the scatter plot we can say that the boosting model has less residuals compared to bagging.

Let us get the R-squared value of the model.

Text, letter

Description automatically generated



Looking at the R-squared value we could say that the boosting model explains about 90% of variance in the model.

1. **Principal Component Analyses (PCA):**

First, let us convert the dataset into lower dimensional data using PCA as below.

A picture containing text

Description automatically generated

Scatter chart

Description automatically generated with medium confidence

Looking at the summary we can say that 95% of the variance in the dataset is explained by the first 12 principal components. Now let us construct a data frame using these 12 components and split the data into train and test data and create a linear model using the train data as below.

Text

Description automatically generated

Now let’s construct a linear model using the training set and evaluate the performance of the model

A picture containing text

Description automatically generated

Text

Description automatically generated

Looking at the summary we can say that the linear model constructed using the PCA explains about 68% of the variance in the data.

Now let us plot a scatter plot of the actual vs the predicted values of the price.

Text, letter

Description automatically generated

Chart, scatter chart

Description automatically generated

Looking at the graph we can say that there are geometry points whose residuals are higher compared to boosting model.

1. **Conclusion**: By comparing the above models, we could say that GAM and Boosting models works well on this dataset, where GAM explains about 83% of the variance in the data and Boosting model explains about 90% of the variance in the data.
2. **Mapview**: Now let us try to visualize the data using a map view, we first get the bounding box of the king county and try to get the map for this bounding box using **stamen** as source and try to render this map using ggmap() and try to project the geometry points on top of this map, for now we try to visualize the most expensive houses (price > 1.5 million) and houses with number of bedrooms greater than 5. Below is the R code that creates the mapview.

Text

Description automatically generated

Map

Description automatically generated

From this map view we can clearly see that the most expensive houses are located nearer to the beach areas.

**Contributions:**

Below are the tasks that we have split among ourselves while working on the project.

* Correlation plot – Meghana Bathula
* Generalized Additive Model - Meghana Bathula
* Bagging – Kapil Pabba
* Boosting – Venkat Velga
* Principal Component Analysis – Kapil Pabba
* Map visualization – Venkat Velga.