PARIS HOUSING
PRICE PREDICTION

Executive Summary

Opportunity:

Paris is renowned for its beauty and cultural richness, yet it's also notorious for its high cost of living, particularly in housing.

The city boasts diverse neighborhoods, each with its unique attributes and associated costs. Its global status draws people from across the globe, resulting in a shortage of housing options and significant affordability challenges. Moreover, stringent government regulations on architectural renovations affect property values.

The housing's location plays a crucial role, with sought-after neighborhoods such as Marais, Saint-Germain-des-Prés, and the Champs-Elysées commanding premium prices, in contrast to the more affordable options found in the outer suburbs.

The pandemic briefly led to a decrease in housing prices. However, starting in 2021, there has been a consistent upward trend. With prices steadily rising over the past decade, there's a growing need to gain a deeper understanding of this trend.

Objective:

Our goal is to provide users with the knowledge to assess the value of Parisian properties by analyzing the factors that impact their prices. To achieve this, we categorize the data into three distinct price ranges using clustering classification. This helps to distinguish the influencing factors more effectively for each segment of the housing market.

Through clustering classification, we assign houses to specific price categories based on factors like size, location, access to amenities, floor level, and more. This process helps us determine whether a property belongs to the luxury, premium, or basic housing sector.

Now that we have an idea about which sector it falls into, we use an appropriate prediction model to estimate a value based on relatable factors according to its market segment of luxury or basic housing and we evaluate the accuracy of each model based on its residuals and performance on test datasets to give an appropriate picture of the pricing structure in the city.

Overall, this project has the potential to revolutionize the real estate industry and provide significant benefits to all stakeholders involved.

Project Motivation:

Living in an inflated market with real estate prices going through the roof, there is an urgent need to understand the influencing factors that may drive the prices.

The real estate industry is a significant contributor to the global economy, and housing prices are influenced by a variety of factors, including location, size, condition, and amenities. Predicting the price of a property accurately is essential for buyers, sellers, and real estate agents, as it can have a significant impact on investment decisions. Traditional methods of pricing properties rely on manual assessment by experts, which can be time-consuming and prone to errors. Therefore, there is a need for a more accurate and automated approach to property valuation.

Machine learning algorithms such as clustering and linear regression have shown promising results in predicting property prices based on a variety of features. Clustering can group similar properties together based on their characteristics, while linear regression can identify the relationship between these features these features and property prices. By combining these two techniques, we can create a robust and accurate model for predicting housing prices.

The goal of this project is to develop a machine learning model that can accurately predict the price of a property based on its features, such as location, size, condition, and amenities. The model will be trained on a Paris housing dataset, and the accuracy of the model will be evaluated using various metrics, such as root mean square error (RMSE), R-squared, and mean absolute error (MAE).

The outcomes of this project will be valuable for a wide range of stakeholders, including buyers, sellers, and real estate agents. Accurate predictions of housing prices will enable buyers to make informed decisions about their investments, and sellers to price their properties competitively in the market. Furthermore, real estate agents can leverage this technology to offer more accurate property valuations to their clients.

Data Description and Analysis:

The housing dataset is a collection of data containing information about various real estate properties in the city of Paris. The dataset consists of information about 10,000 houses including 17 metrics about area, amenities, number of previous owners etc. with each sample representing a unique property as well as their corresponding prices. The following is a description of the data features:

- *squareMeters* - Area in square meters of the house.
- *numberOfRooms* - Number of Rooms in the house.
- *hasYard* - Has a Yard or not (Categorical)
- *hasPool* - Has Pool or not (Categorical)
- *floors* - What floor the house is on
- *cityCode* - Area code depicting the area in which the property is located.
- *cityPartRange* - the higher the range, the more exclusive the neighborhood is (from 0-10, cheapest to most expensive)
- *numPrevOwners* - number of previous owners

- *made* - year the house has been built
- *isNewBuilt* - Is the house new or renovated? (Categorical)
- *hasStormProtector* - Does the house have a storm protector? (Categorical)
- *basement* - basement area in square meters.
- *attic* - attic area in square meters.
- *garage* - garage area in square meters.
- *hasStorageRoom* - Whether it has storage space (Categorical)
- *hasGuestRoom* - Number of guest rooms.

Steps to verifying the Dataset:

1. Verifying the summary of the structure of the dataset to understand the data values, whether we need to change some variables to factors or normalize the data.

```
> ParisHousing <- read.csv("C:/Users/nambi/Downloads/PARIS HOUSING PRICE/ParisHousing.csv")
>   View(ParisHousing)
> str(ParisHousing)
'data.frame':   10000 obs. of  17 variables:
 $ squareMeters     : int  75523 80771 55712 32316 70429 39223 58682 86929 51522 39686 ...
 $ numberOfRooms    : int  3 39 58 47 19 36 10 100 3 42 ...
 $ hasYard          : int  0 1 0 0 1 0 1 1 0 0 ...
 $ hasPool          : int  1 1 1 0 1 1 1 0 0 0 ...
 $ floors           : int  63 98 19 6 90 17 99 11 61 15 ...
 $ cityCode         : int  9373 39381 34457 27939 38045 39489 6450 98155 9047 71019 ...
 $ cityPartRange    : int  3 8 6 10 3 8 10 3 8 5 ...
 $ numPrevOwners    : int  8 6 8 4 7 6 9 4 3 8 ...
 $ made             : int  2005 2015 2021 2012 1990 2012 1995 2003 2012 2021 ...
 $ isNewBuilt       : int  0 1 0 0 1 0 1 1 1 1 ...
 $ hasStormProtector: int  1 0 0 1 0 1 1 0 1 1 ...
 $ basement         : int  4313 3653 2937 659 8435 2009 5930 6326 632 5198 ...
 $ attic            : int  9005 2436 8852 7141 2429 4552 9453 4748 5792 5342 ...
 $ garage           : int  956 128 135 359 292 757 848 654 807 591 ...
 $ hasStorageRoom   : int  0 1 1 0 1 0 0 0 1 1 ...
 $ hasGuestRoom     : int  7 2 9 3 4 1 5 10 5 3 ...
 $ price            : num  7559082 8085990 5574642 3232561 7055052 ...
```

2. Next, we verify that there are no missing values or major outliers in the data that may affect the model outputs significantly.

```
> sum(is.na (ParisHousing))
[1] 0
>
```

3. Checking the summary statistics in the data to get insights into the variability of the individual features.

```
> summary(ParisHousing)
  squareMeters   numberOfRooms       hasYard          hasPool           floors
 Min.   :   89   Min.   :  1.00   Min.   :0.0000   Min.   :0.0000   Min.   :  1.00
 1st Qu.:25099   1st Qu.: 25.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 25.00
 Median :50106   Median : 50.00   Median :1.0000   Median :0.0000   Median : 50.00
 Mean   :49870   Mean   : 50.36   Mean   :0.5087   Mean   :0.4968   Mean   : 50.28
 3rd Qu.:74610   3rd Qu.: 75.00   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 76.00
 Max.   :99999   Max.   :100.00   Max.   :1.0000   Max.   :1.0000   Max.   :100.00
    cityCode     cityPartRange   numPrevOwners        made        isNewBuilt
 Min.   :    3   Min.   : 1.00   Min.   : 1.000   Min.   :1990   Min.   :0.0000
 1st Qu.:24694   1st Qu.: 3.00   1st Qu.: 3.000   1st Qu.:1997   1st Qu.:0.0000
 Median :50693   Median : 5.00   Median : 5.000   Median :2006   Median :0.0000
 Mean   :50225   Mean   : 5.51   Mean   : 5.522   Mean   :2005   Mean   :0.4991
 3rd Qu.:75683   3rd Qu.: 8.00   3rd Qu.: 8.000   3rd Qu.:2014   3rd Qu.:1.0000
 Max.   :99953   Max.   :10.00   Max.   :10.000   Max.   :2021   Max.   :1.0000
 hasStormProtector    basement         attic          garage       hasStorageRoom
 Min.   :0.0000    Min.   :    0   Min.   :    1   Min.   : 100.0   Min.   :0.000
 1st Qu.:0.0000    1st Qu.: 2560   1st Qu.: 2512   1st Qu.: 327.8   1st Qu.:0.000
 Median :0.0000    Median : 5092   Median : 5045   Median : 554.0   Median :1.000
 Mean   :0.4999    Mean   : 5033   Mean   : 5028   Mean   : 553.1   Mean   :0.503
 3rd Qu.:1.0000    3rd Qu.: 7511   3rd Qu.: 7540   3rd Qu.: 777.2   3rd Qu.:1.000
 Max.   :1.0000    Max.   :10000   Max.   :10000   Max.   :1000.0   Max.   :1.000
  hasGuestRoom        price
 Min.   : 0.000   Min.   :   10314
 1st Qu.: 2.000   1st Qu.: 2516402
 Median : 5.000   Median : 5016180
 Mean   : 4.995   Mean   : 4993448
 3rd Qu.: 8.000   3rd Qu.: 7469092
 Max.   :10.000   Max.   :10006771
```

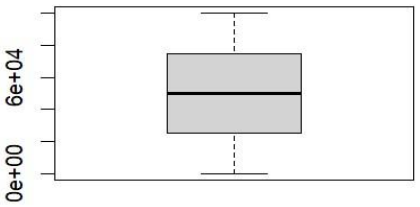
4. Next, we check the correlation between the variables to understand if we need to remove highly correlated factors to create a better model.

```
> cor(ParisHousing)
                  squareMeters numberOfRooms       hasYard       hasPool        floors
squareMeters       1.0000000000   0.009572776 -0.0066498548 -0.0055943354  0.0011093139
numberOfRooms      0.0095727760   1.000000000 -0.0112400597  0.0170154244  0.0222442495
hasYard           -0.0066498548  -0.011240060  1.0000000000  0.0155140264 -0.0008831253
hasPool           -0.0055943354   0.017015424  0.0155140264  1.0000000000 -0.0040063398
floors             0.0011093139   0.022244250 -0.0008831253 -0.0040063398  1.0000000000
cityCode          -0.0015405537   0.009039650  0.0067599035  0.0080719802  0.0022073544
cityPartRange      0.0087582201   0.008340145  0.0050233440  0.0146125504 -0.0049208087
numPrevOwners      0.0166186661   0.016765862  0.0042794217 -0.0068480156  0.0024631365
made              -0.0072071480   0.003978432  0.0022135852  0.0018937981  0.0050217458
isNewBuilt        -0.0106671233  -0.002864976 -0.0083699607  0.0001884842  0.0024577031
hasStormProtector  0.0074800370  -0.001656356 -0.0075976704 -0.0010013005 -0.0085657396
basement          -0.0039602838  -0.013990016 -0.0085580745 -0.0072677352  0.0062278341
attic             -0.0005880379   0.012060636 -0.0030849197 -0.0119006242 -0.0002704207
garage            -0.0172459023   0.023187780 -0.0046256524  0.0048321469  0.0113031189
hasStorageRoom    -0.0034862897  -0.004759731 -0.0095060102  0.0012384477  0.0036155174
hasGuestRoom      -0.0006229538  -0.015528791 -0.0072757279  0.0011225539 -0.0211554128
price              0.9999993571   0.009590906 -0.0061192449 -0.0050703408  0.0016542562
                     cityCode cityPartRange numPrevOwners          made    isNewBuilt
squareMeters      -0.001540554   0.008758220  0.0166186661 -0.0072071480 -0.0106671233
numberOfRooms      0.009039650   0.008340145  0.0167658622  0.0039784316 -0.0028649759
hasYard            0.006759904   0.005023344  0.0042794217  0.0022135852 -0.0083699607
hasPool            0.008071980   0.014612550 -0.0068480156  0.0018937981  0.0001884842
floors             0.002207354  -0.004920809  0.0024631365  0.0050217458  0.0024577031
cityCode           1.000000000   0.011333585 -0.0075492829  0.0092663402 -0.0002239540
cityPartRange      0.011333585   1.000000000  0.0092375141  0.0077483415 -0.0018739737
numPrevOwners     -0.007549283   0.009237514  1.0000000000  0.0068584817 -0.0174201307
made               0.009266340   0.007748341  0.0068584817  1.0000000000 -0.0016782720
isNewBuilt        -0.000223954  -0.001873974 -0.0174201307 -0.0016782720  1.0000000000
hasStormProtector -0.004941496   0.005223762  0.0025220654 -0.0006448799  0.0031996452
basement           0.002652441   0.004742864 -0.0008617590 -0.0055056425 -0.0159864479
```

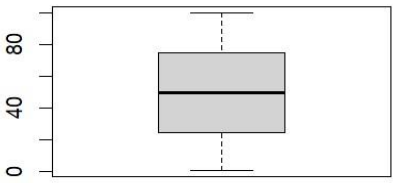
```
attic            -0.002019391   0.010695795  0.0007185488  0.0137726406  0.0201265169
garage           -0.002208117  -0.001647920  0.0202675521  0.0056869293  0.0027493031
hasStorageRoom    0.002554254  -0.011337932  0.0317068688 -0.0078679926  0.0070109376
hasGuestRoom     -0.003338363  -0.007152685 -0.0060820669 -0.0054311197  0.0198946434
price            -0.001539367   0.008812912  0.0166188261 -0.0072095263 -0.0106427744
```

5. To evaluate whether there are any significant outliers in the data we create boxplots for the variables in the dataset.

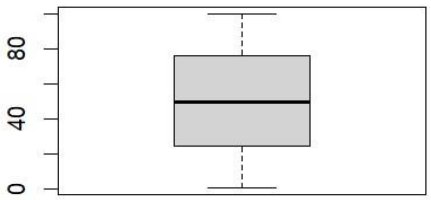
```
> boxplot (ParisHousing)
> for (col in names (ParisHousing)) {
+   boxplot (ParisHousing [col], xlab = col)
+ }
> d <- dist (housing_data, method "euclidean")
```



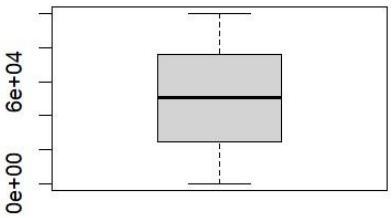
squareMeters



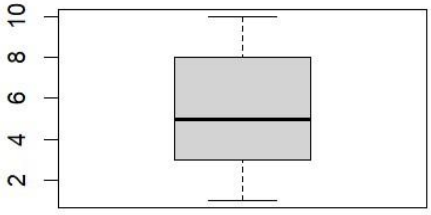
numberOfRooms



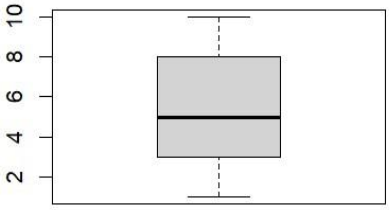
floors



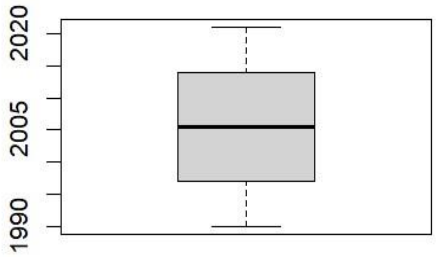
cityCode



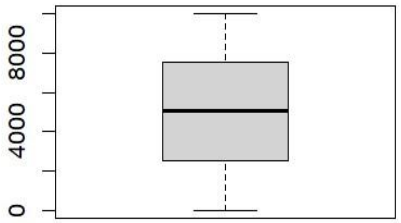
cityPartRange



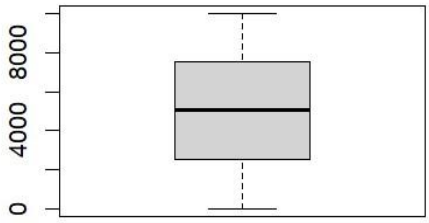
numPrevOwners



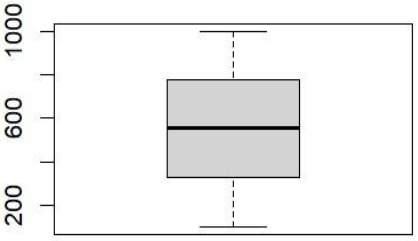
made



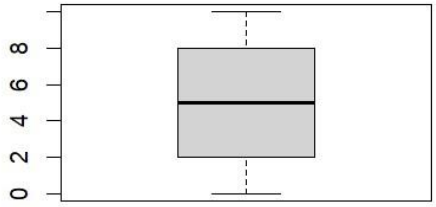
basement



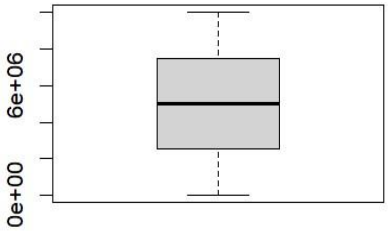
attic



garage



hasGuestRoom



price

BI Model

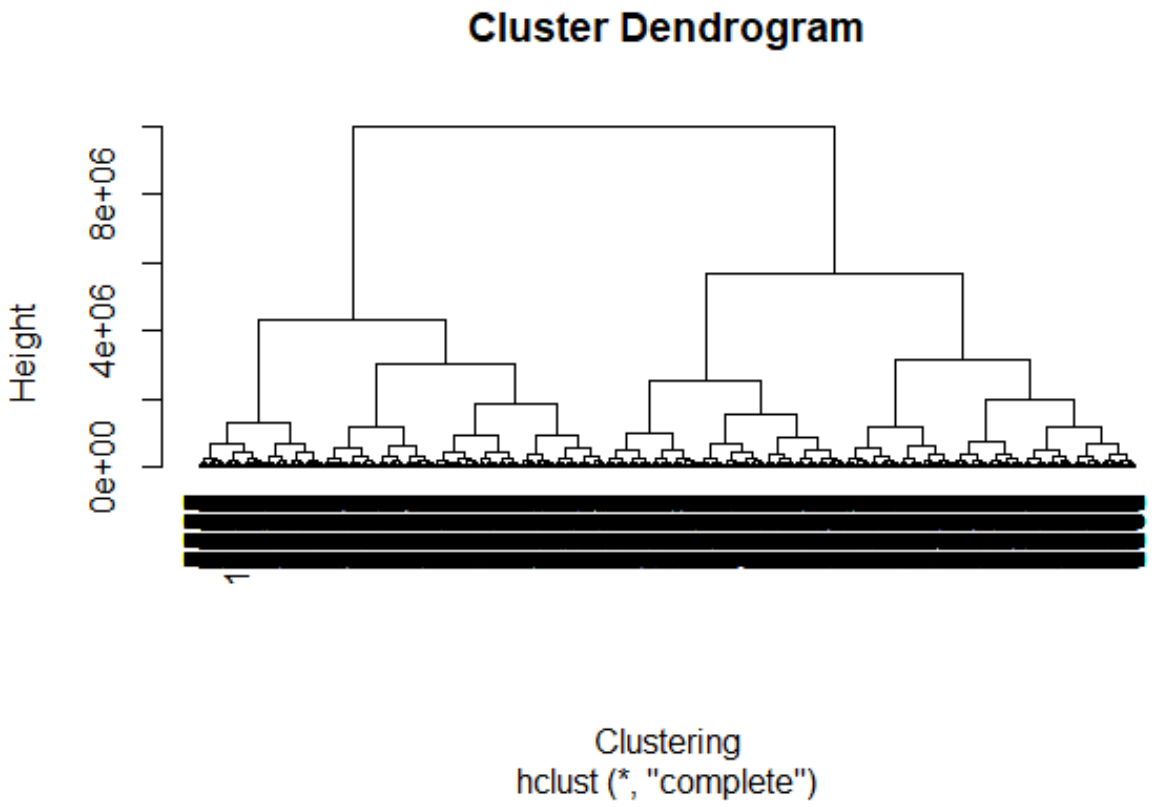
Clustering:

While predicting the housing prices in Paris, we essentially use a two-step approach - Clustering and a Linear Regression Model.

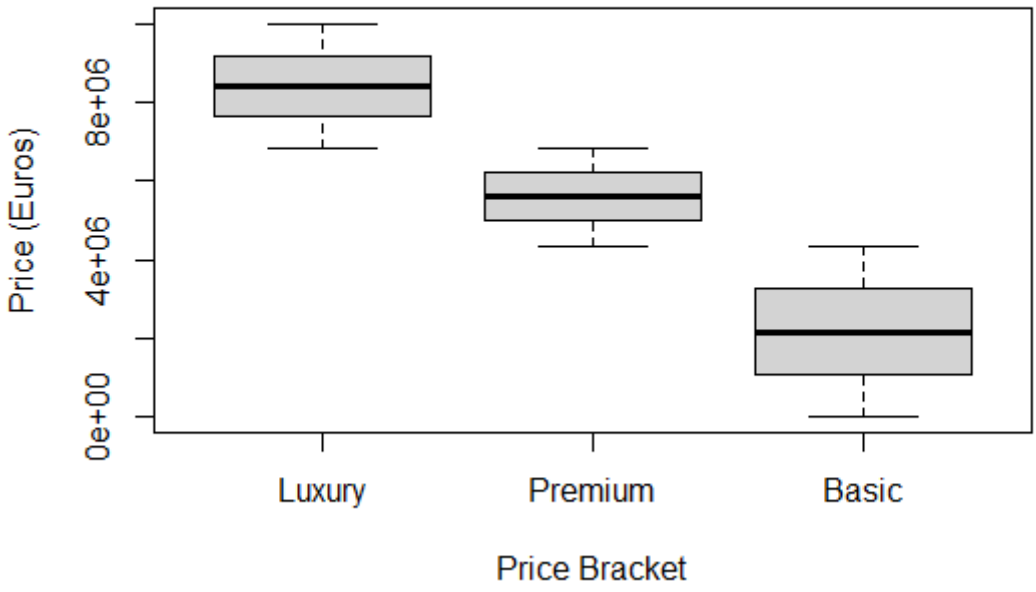
Since the house pricing ranges from 10,000 Euros to 10 million euros, we are first dividing the dataset into three price brackets, luxury, premium and basic housing.

To achieve this, we use the K-nearest neighbor (KNN) clustering method to divide the dataset based on the independent variables relating to area, size, amenities and condition of the house and previous ownership. We use the original dataset here to have a better estimate of the clusters to segregate it into different price brackets.

```
> # Calculate the pairwise Euclidean distances
> d <- dist(ParisHousing, method = "euclidean")
> # Perform hierarchical clustering on the distance matrix
> hc_withoutnorm <- hclust(d, method = "complete")
> # Plot a dendrogram of the hierarchical clustering results
> plot(hc_withoutnorm, hang = -1, ann = T, xlab = "Clustering")
```



This gives us the dendrogram of the clustering analysis. Now since we are dividing the dataset into three price brackets, we cut the tree into three clusters and name them accordingly.



Now we have the clustering analysis done, we will predict prices using linear regression for each price bracket while improving the model accuracy by removing the insignificant variables.

Linear regression Model

Luxury Housing Price Prediction

We create the luxury housing subset from the data and remove the insignificant variables in the data for an accurate model:

```
> luxury <- subset (ParisHousing, pricebracket == "Luxury")
> luxury <- luxury [-c(18)] #Removing the price bracket variable
> View(luxury)
> luxury <- luxury[-c(2,6,8,9,12,13,14,15)]#Removing insignificant variabl
es
```

Next, we create the training and test datasets:

```
> nrows <- nrow (luxury)
> sample.index <- sample(c(1:nrows), 0.8*nrows)
> luxhousing_train <- luxury [sample.index,]
> luxhousing_test <- luxury[-sample.index,]
```

With the datasets ready for analysis, we run the linear regression model and understand the significant variables and remove the other variables as mentioned earlier.

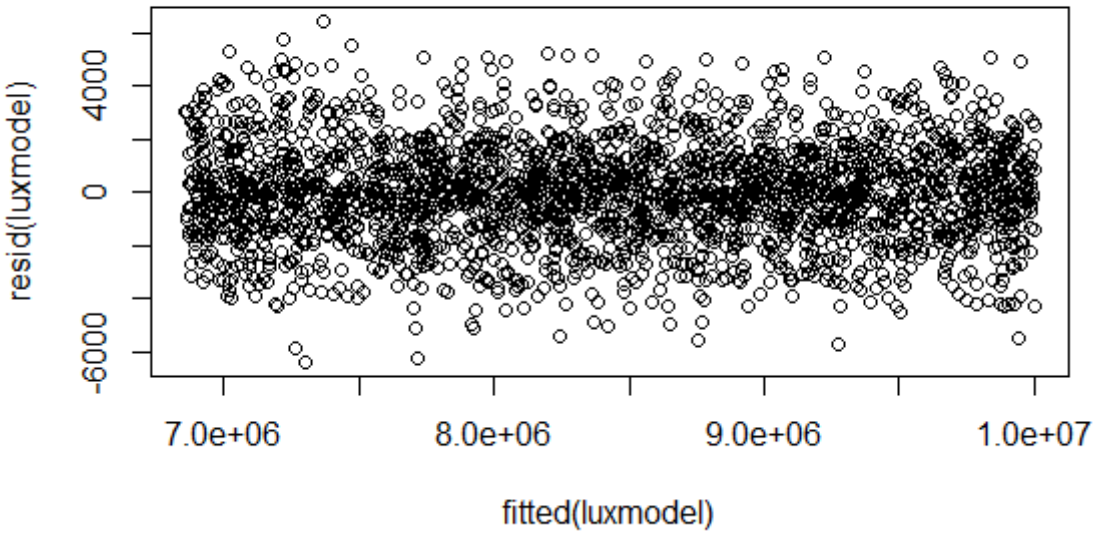
```
> luxmodel <- lm(price ~., luxhousing_train)
> Summary(luxmodel)
```

```
Call:
lm(formula = price ~ ., data = luxhousing_train)

Residuals:
    Min      1Q  Median      3Q     Max
-6359.2 -1195.0   -11.4  1162.4  6423.0

Coefficients:
                   Estimate Std. Error   t value Pr(>|t|)
(Intercept)        7.335e+02  3.755e+02     1.954  0.05087 .
squareMeters       9.999e+01  4.148e-03 24106.220  < 2e-16 ***
hasYard            3.023e+03  7.561e+01    39.985  < 2e-16 ***
hasPool            3.011e+03  7.550e+01    39.883  < 2e-16 ***
floors             5.463e+01  1.309e+00    41.743  < 2e-16 ***
cityPartRange      4.169e+01  1.329e+01     3.137  0.00173 **
isNewBuilt         1.827e+02  7.549e+01     2.420  0.01559 *
hasStormProtector  1.252e+02  7.561e+01     1.656  0.09789 .
hasGuestRoom      -8.813e+00  1.206e+01    -0.731  0.46507
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1879 on 2477 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:      1
F-statistic: 7.279e+07 on 8 and 2477 DF,  p-value: < 2.2e-16
```

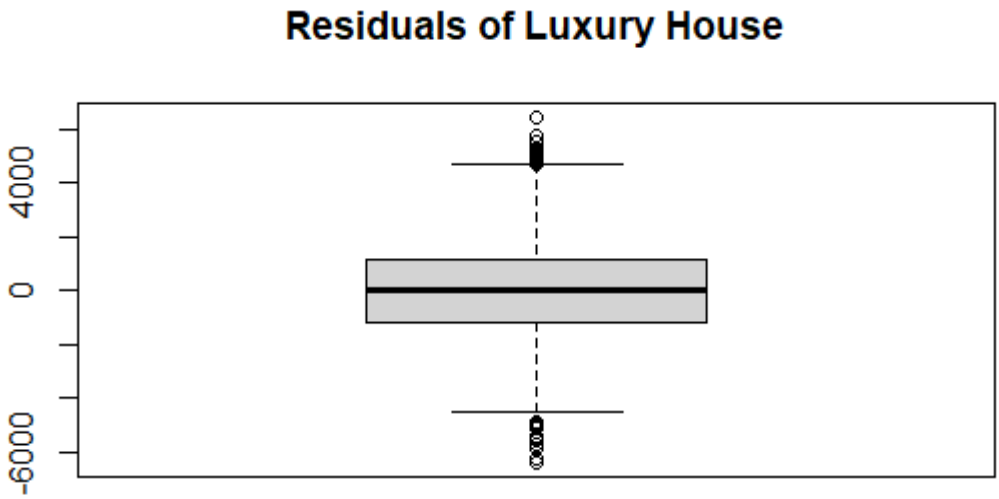
As we can see in the model summary above, we have a model with all the variables used in significance and contributing to the price.

The variables squareMeters, hasYard, hasPool, floors are the most significant and cityPartRange, isNewBuilt and hasGuestRoom have lesser significance for the price prediction.
Now we find insights regarding the residuals to evaluate model accuracy.

```
> boxplot(resid(luxmodel))
> boxplot(resid(luxmodel),main = "Residuals of Luxury House")
```

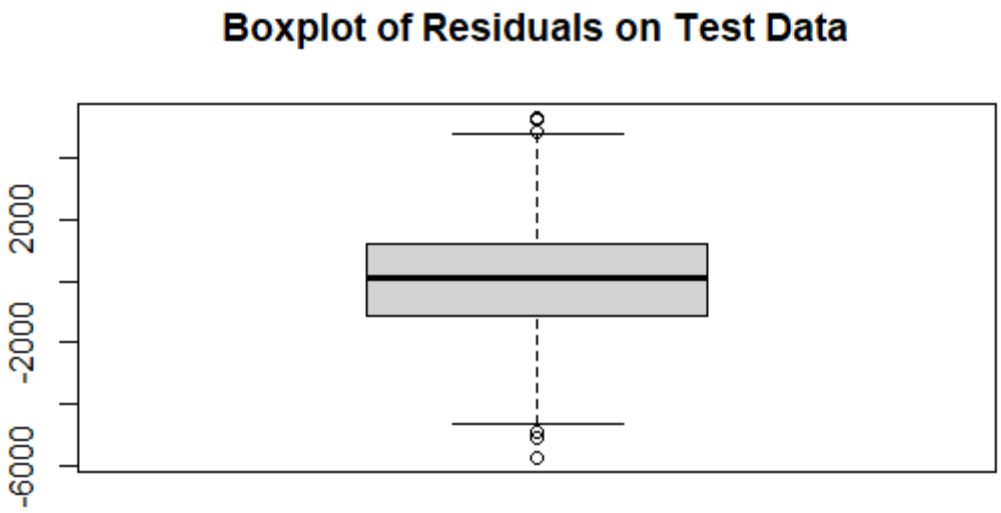


Here is the Box plot for the Residuals of Luxury Housing Proce Model



With the model ready, we can now use it to predict the prices for the test dataset and check its accuracy.

```
> # Generating predictions on the test set
> luxhousing_testPrediction <- predict(luxmodel, newdata = luxhousing_test
)
> # Calculating residuals
> residuals_test <- luxhousing_test$price - luxhousing_testPrediction
> # Creating a boxplot for residuals
> boxplot(residuals_test, main = "Boxplot of Residuals on Test Data")
```



Premium Housing Price Prediction

We create the premium housing subset from the data and remove the insignificant variables in the data for an accurate model:

```
> premium <- subset(ParisHousing, pricebracket == "Premium")
> premium <- premium[-c(18)]
> View(premium)
> premium <- premium[-c(8, 12, 13, 14, 15)]
```

Next we create the training and test datasets:

```
> nrows <- nrow(premium)
> sample.index <- sample(c(1:nrows), 0.8 * nrows)
> premiumhousing_train <- premium[sample.index,]
> premiumhousing_test <- premium[-sample.index,]
```

With the datasets ready for analysis, we run the linear regression model and understand the significant variables and remove the other variables as mentioned earlier.

```
> premiummodel <- lm(price ~ ., premiumhousing_train)
> summary(premiummodel)
```

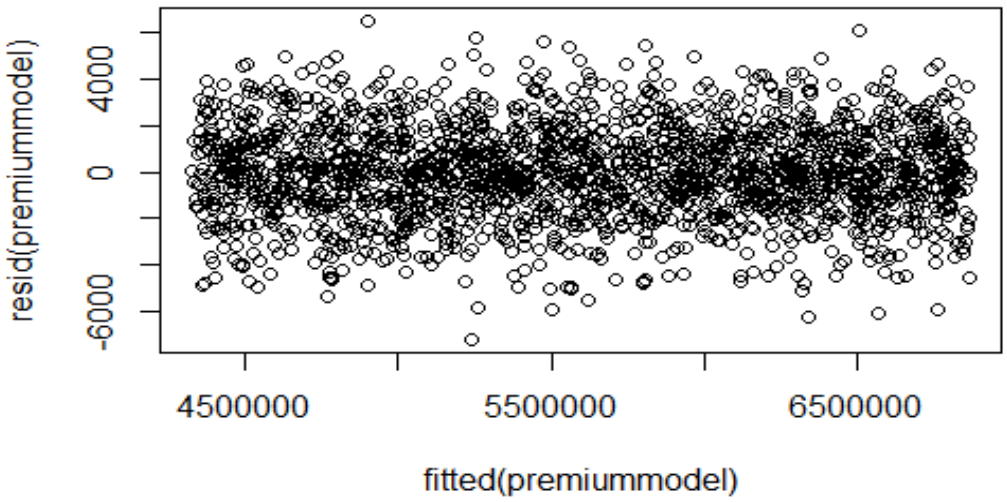
```
Call:
lm(formula = price ~ ., data = premiumhousing_train)

Residuals:
    Min      1Q  Median      3Q     Max
-7199.0 -1240.5   -18.3  1288.7  6520.6

Coefficients:
                  Estimate Std. Error   t value Pr(>|t|)
(Intercept)       1.052e+04  9.307e+03     1.130 0.258633
squareMeters      9.999e+01  5.955e-03 16791.603  < 2e-16 ***
numberOfRooms    -1.549e+00  1.525e+00    -1.016 0.309893
hasYard           2.930e+03  8.693e+01    33.706  < 2e-16 ***
hasPool           3.010e+03  8.691e+01    34.634  < 2e-16 ***
floors            5.473e+01  1.526e+00    35.866  < 2e-16 ***
cityCode          4.078e-04  1.487e-03     0.274 0.783891
cityPartRange     5.439e+01  1.507e+01     3.609 0.000314 ***
made             -5.005e+00  4.639e+00    -1.079 0.280759
isNewBuilt        3.089e+02  8.701e+01     3.550 0.000393 ***
hasStormProtector 7.535e+01  8.690e+01     0.867 0.386001
hasGuestRoom      2.575e+01  1.380e+01     1.866 0.062204 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1959 on 2027 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:      1
F-statistic: 2.571e+07 on 11 and 2027 DF,  p-value: < 2.2e-16
```

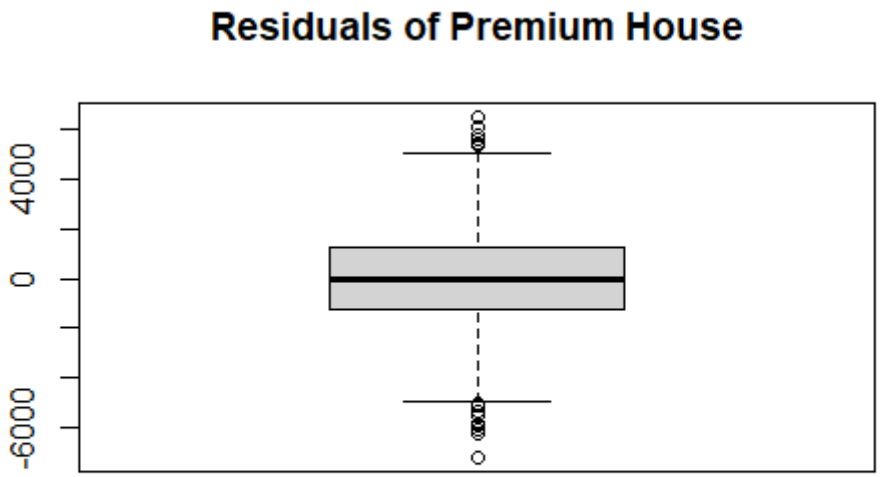
As we can see in the model summary above, we have a model with all the variables used in significance and contributing to the price.

The variables squareMeters, hasYard, hasPool, floors, isNewBuilt are the most significant and cityPartRange and cityCode have lesser significance for the price prediction.

Now we find insights regarding the residuals to evaluate model accuracy.

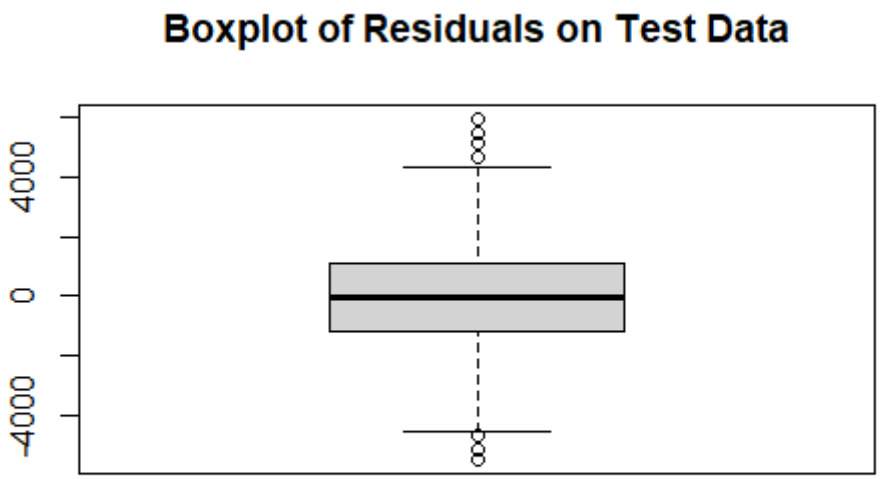
```
> plot(fitted(premiummodel), resid(premiummodel))
> boxplot(resid(premiummodel), main = "Residuals of Premium House")
```





With the model ready, we can now use it to predict the prices for the test dataset and check its accuracy.

```
> premiumhousing_testPrediction <- predict(premiummodel, newdata = premium
housing_test)
> residuals_test <- premiumhousing_test$price - premiumhousing_testPredict
ion
> boxplot(residuals_test, main = "Boxplot of Residuals on Test Data")
```



Basic Housing Price Prediction

We create the basic housing subset from the data and remove the insignificant variables in the data for an accurate model:

```
> basic <- subset(ParisHousing, pricebracket == "Basic")
> basic <- basic[-c(18)]
> View(basic)
> basic <- basic[-c(8)]
```

Next, we create the training and test datasets:

```
> nrows <- nrow(basic)
> sample.index <- sample(c(1:nrows), 0.8 * nrows)
> basichousing_train <- basic[sample.index,]
> basichousing_test <- basic[-sample.index,]
```

With the datasets ready for analysis, we run the linear regression model and understand the significant variables and remove the other variables as mentioned earlier.

```
> basicmodel <- lm(price ~ ., basichousing_train)
> summary(basicmodel)
```

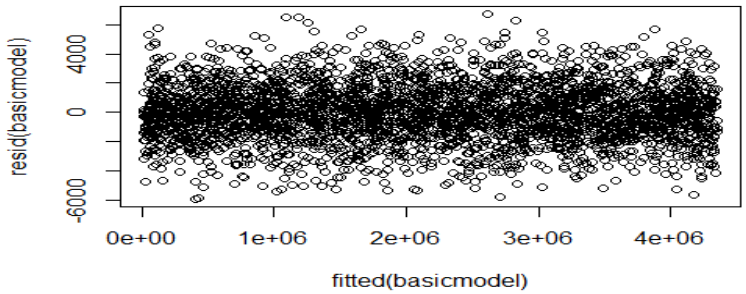
```
Call:
lm(formula = price ~ ., data = basichousing_train)

Residuals:
   Min     1Q Median     3Q    Max
 -5893  -1198    -23   1181   6713

Coefficients:
                    Estimate Std. Error   t value Pr(>|t|)
(Intercept)        1.081e+04  6.854e+03     1.578 0.114738
squareMeters       1.000e+02  2.530e-03 39520.824  < 2e-16 ***
numberOfRooms      1.311e+00  1.102e+00     1.190 0.234164
hasYard            2.962e+03  6.425e+01    46.107  < 2e-16 ***
hasPool            2.959e+03  6.432e+01    46.004  < 2e-16 ***
floors             5.295e+01  1.106e+00    47.878  < 2e-16 ***
cityCode          -2.202e-03  1.108e-03    -1.988 0.046878 *
cityPartRange      5.249e+01  1.114e+01     4.712 2.55e-06 ***
made              -5.265e+00  3.419e+00    -1.540 0.123629
isNewBuilt         6.249e+01  6.427e+01     0.972 0.330946
hasStormProtector  2.166e+02  6.428e+01     3.370 0.000761 ***
basement           8.094e-03  1.105e-02     0.733 0.463871
attic             -7.573e-03  1.099e-02    -0.689 0.491015
garage             1.594e-01  1.229e-01     1.297 0.194769
hasStorageRoom     6.063e+01  6.432e+01     0.943 0.345949
hasGuestRoom       2.720e+00  1.004e+01     0.271 0.786401
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

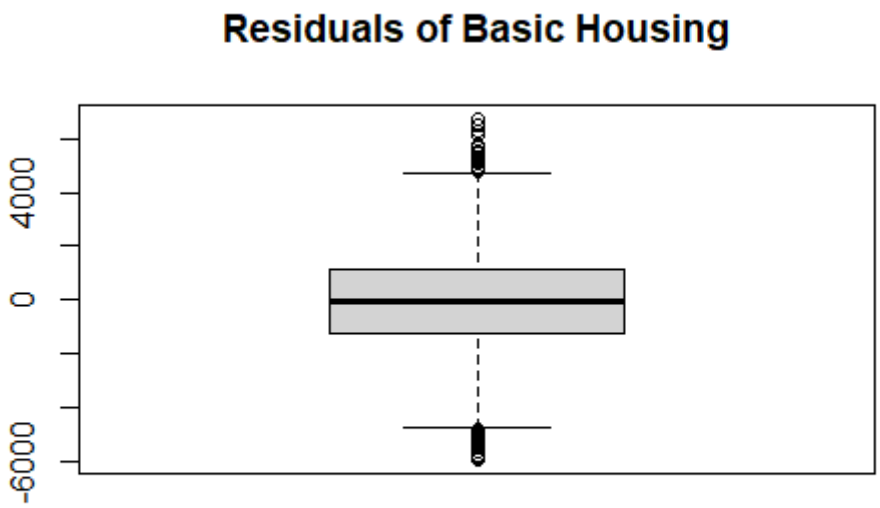
As we can see in the model summary above, we have a model with all the variables used in significance and contributing to the price.

The variables squareMeters, hasYard, hasPool, floors are the most significant and hasStormProtector and garage have lesser significance for the price prediction.

Now we find insights regarding the residuals to evaluate model accuracy.

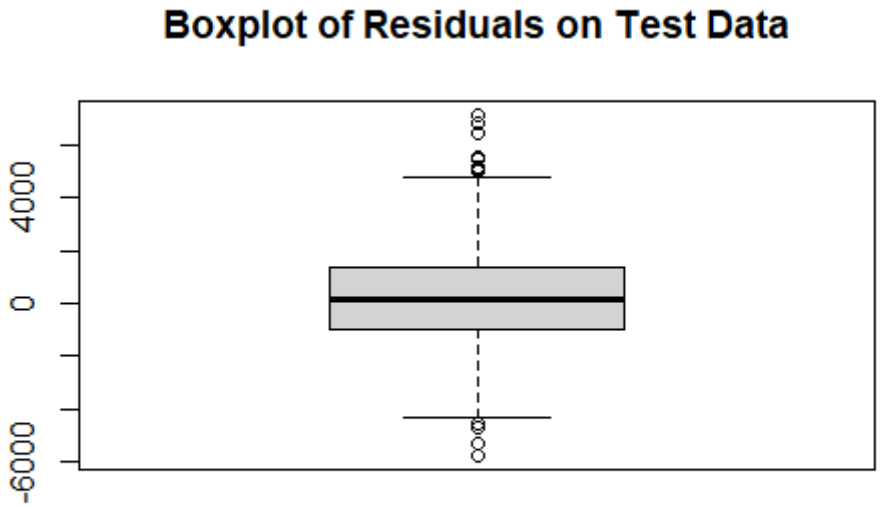
```
> plot(fitted(basicmodel), resid(basicmodel))
> boxplot(resid(basicmodel), main = "Residuals of Basic Housing")
```





With the model ready, we can now use it to predict the prices for the test dataset and check its accuracy.

```
> basichousing_testPrediction <- predict(basicmodel, newdata = basichousin
g_test)
> residuals_test <- basichousing_test$price - basichousing_testPrediction
> boxplot(residuals_test, main = "Boxplot of Residuals on Test Data")
```



Decision Tree

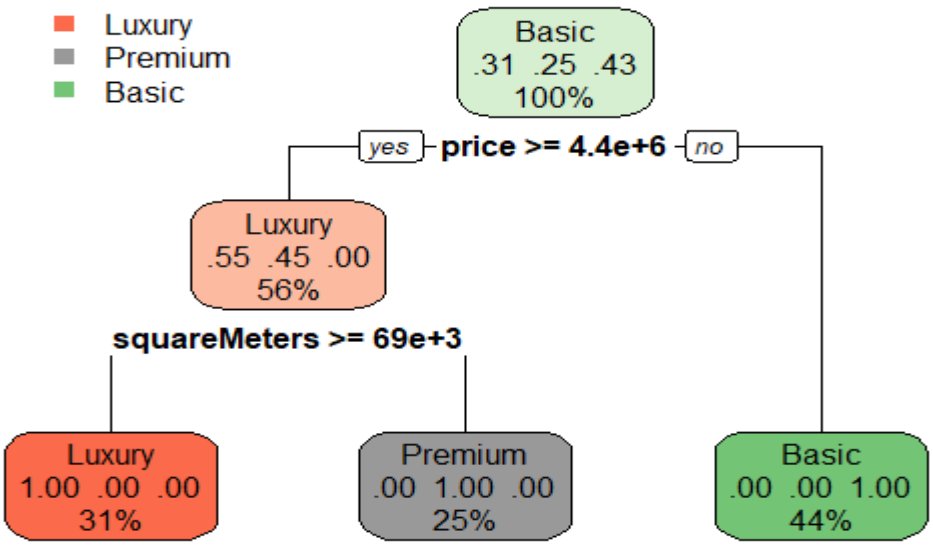
Interpreting the Decision Tree

The decision tree generated for our Paris housing dataset helps us comprehend the hierarchy of features influencing the categorization of properties into different price brackets. Each node in the tree represents a decision point based on a particular attribute, leading to further branching based on its value.

- **Root Node:** The initial split that maximizes the distinction between different price brackets.
- **Internal Nodes:** Subsequent splits based on different property attributes (e.g., square meters, number of rooms, location).
- **Leaf Nodes:** Final outcomes or classifications (Luxury, Premium, Basic) after traversing through the tree.

We utilized the **rpart** package in R, which provides an efficient implementation of decision trees.

```
> library(rpart)
> library(rpart.plot)
> # Assuming you have a dataframe 'data' with predictor variables and a ta
rget variable 'target'
> # Create a decision tree model
> tree_model <- rpart(pricebracket ~ ., data = ParisHousing, method = "cla
ss")
> # Plot the decision tree
> rpart.plot(tree_model)
```



Decision trees provide a clear and intuitive representation of the factors influencing the target variable (in this case, housing price brackets). This visual representation enables stakeholders to comprehend complex relationships between various attributes and the target variable, fostering better insight.

Neural Network

Neural networks are composed of layers of interconnected nodes, or "neurons," each of which performs a simple computation. Data flows through these layers, being transformed at each step according to the parameters of the network (the weights and biases of the connections). In R, you define these networks using the aforementioned packages, specifying the architecture (number of layers, number of neurons in each layer, activation functions, etc.) and training parameters.

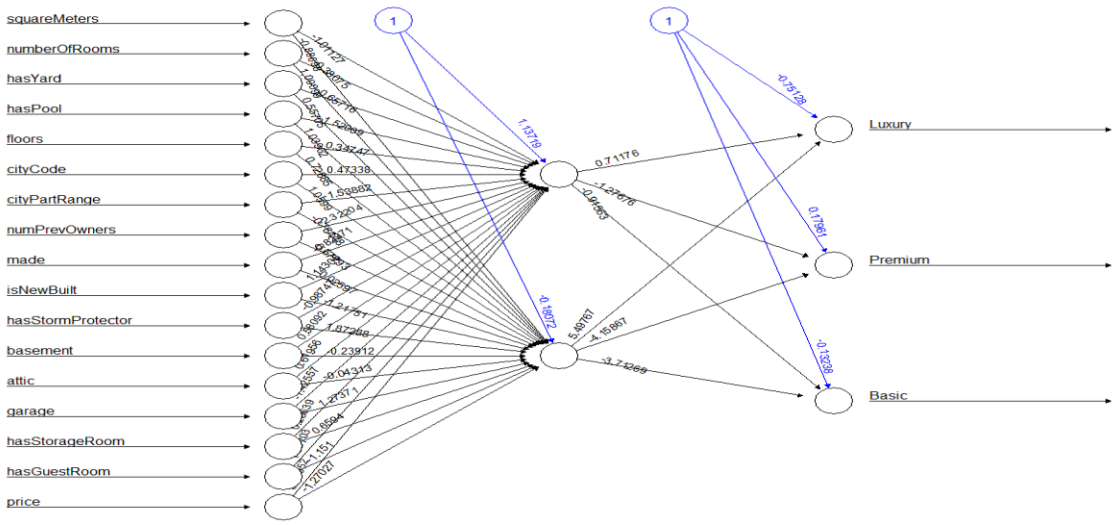
Network Topology:

Input Layer: The input layer has multiple nodes, each representing a feature that influences housing prices, such as 'squareMeters', 'numberOfRooms', 'hasYard', 'hasPool', 'floors', etc. These features are the raw inputs that feed data into the network.

Hidden Layer: There is at least one hidden layer with multiple neurons (the number of neurons is not fully visible in the image). The hidden layer transforms the inputs into a space where the output variable can be predicted with more accuracy. Each neuron in the hidden layer is connected to all input nodes, with each connection having a weight (these are the numbers close to the lines). The weights determine the strength and direction of the influence of an input node on a neuron.

Output Layer: The output layer consists of three nodes: 'Luxury', 'Premium', and 'Basic'. This suggests that the neural network is configured for a classification task to categorize houses into one of these three categories based on the input features.

```
> library(neuralnet)
> nn <- neuralnet(pricebracket~.,data = housing,linear.output = TRUE,hidde
n = c(2))
> plot(nn)
```



The neural network shown is a feedforward neural network, which is typically used for supervised learning tasks. In this case, it seems to be used for a classification problem, categorizing houses into 'Luxury', 'Premium', or 'Basic' based on their features.

Architecture: The model's architecture includes an input layer with several nodes representing different housing features, at least one hidden layer with multiple neurons, and an output layer with three nodes corresponding to the classification categories.

Analytics Insights and Suggested Strategy:

Deciphering the outputs of the analytics performed in the project, we observe that dividing the housing into brackets of luxury, premium and basic housing helps us determine the specific factors that may determine the price for a real estate.

A real estate agent should use these insights to better determine the sale value for property based on what price bracket it lies in. On the other hand, a potential buyer should use this analysis to better understand the pricing reasoning and how certain factors such as area, city locality and other amenities may increase or decrease the valuation of a property.

For a luxury housing unit, the area of the house, the construction modernity as well as luxury amenities such as a pool and what floor in a high rise is the unit in would drive the price upwards. Also having the house located in an ideal locality would increase the valuation.

For a premium housing unit, similar to a luxury housing unit, the square footage, recent construction, having yard space as well as being in a posh district would drive the price. And for the basic housing sector, apart from the area again being the driving factor, the floor, the house is located in as well as necessary amenities such as garage space and having a storm protector for safety against harsh weather conditions will entice a potential buyer to be ready to pay a higher price for an ideal stay.

The neural network depicted in the image is designed to classify housing listings into different quality categories. It is a clear example of how machine learning can be applied to real estate to add value to both buyers and sellers by providing automated insights into property classifications.

In conclusion, this analysis empowers real-estate personnel to make better pricing decisions based on important factors that would satisfy a consumer and a potential buyer be able to understand the reasons for whether a house is well priced or not according to the amenities provided and judge a lease contract better using this predicted price estimate.

References

https://www.kaggle.com/datasets/mssmartypants/paris-housing-price-prediction

https://www.sightline.org/2021/07/26/yes-other-places-do-housing-better-case-3-paris/

https://www.statista.com/statistics/744823/square-meter-price-housing-by-type-france

https://robbreport.com/shelter/homes-for-sale/paris-luxury-home-prices-1234758899/

https://parispropertygroup.com/blog/2018/5-year-review-of-paris-real-estate-price-evolution/