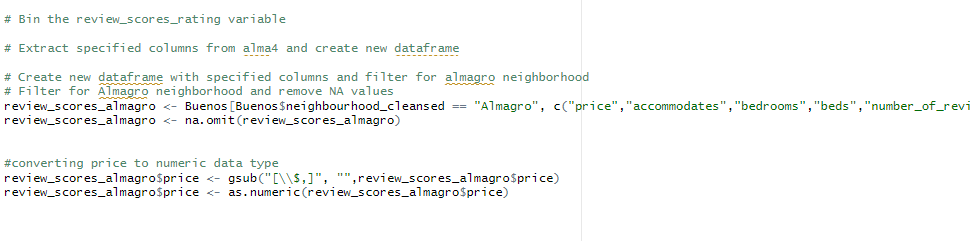
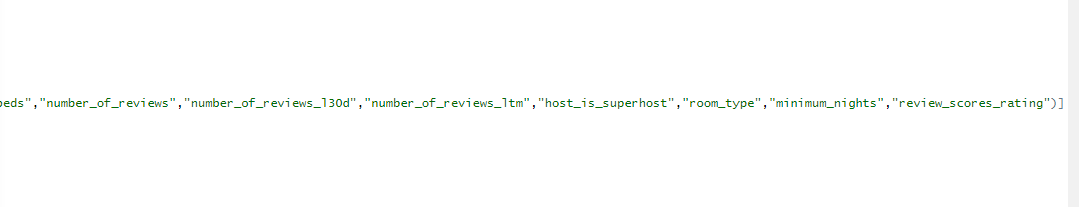
**Classification trees**

****

****

From the Almagro dataset , the input variables that are related to review scores are –

The variables listed here are likely related to review scores in the following ways:

Price: A higher-priced listing may be expected to offer more luxurious amenities and features, and guests may therefore have higher expectations for their stay. If a listing is priced too high and fails to meet these expectations, it could result in lower review scores.

Accommodates, bedrooms, and beds: These variables are all related to the size and capacity of the listing. If a listing is advertised as being able to accommodate a certain number of guests, but does not have enough space or beds to comfortably do so, it could result in lower review scores.

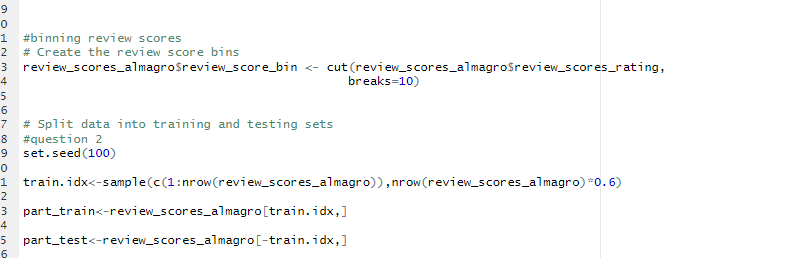
Number of reviews and number of reviews in the last 30 days: Listings with a high number of reviews, particularly recent reviews, may be seen as more trustworthy and reliable, and could therefore have higher review scores.

Host is superhost: Being a superhost means that the host has a track record of providing excellent service and hospitality to guests. Listings hosted by superhosts may therefore be more likely to receive higher review scores.

Room type: The type of room (e.g. private room, entire home/apartment, shared room) can impact guests' expectations and experiences. For example, guests staying in a shared room may have different expectations and experiences than those staying in an entire home or apartment.

Minimum nights: The length of the minimum stay can impact guests' experiences and their overall satisfaction with the listing. For example, guests who are required to stay a minimum of 7 nights may have different expectations and experiences than those who are only required to stay a minimum of 2 nights.

Review scores rating: This variable is itself a measure of the listing's overall review scores, so it is likely to be strongly related to review scores in general.

****

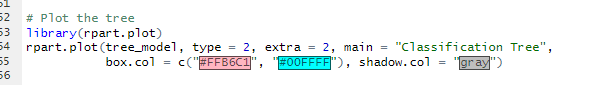
This code performs the following actions:

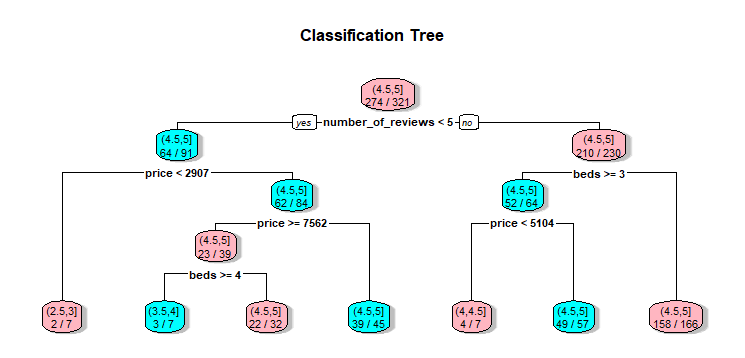
Binning the review scores: It creates a new column called "review\_score\_bin" in the "review\_scores\_almagro" dataframe by categorizing the review\_scores\_rating variable into 10 equal-width bins using the cut function.

Splitting the data into training and testing sets: It sets the seed to 100 for reproducibility, and randomly samples 60% of the rows in the "review\_scores\_almagro" dataframe to use as the training set. The remaining 40% is used as the test set, and saved as "part\_train" and "part\_test", respectively.

Then the tree model is created -

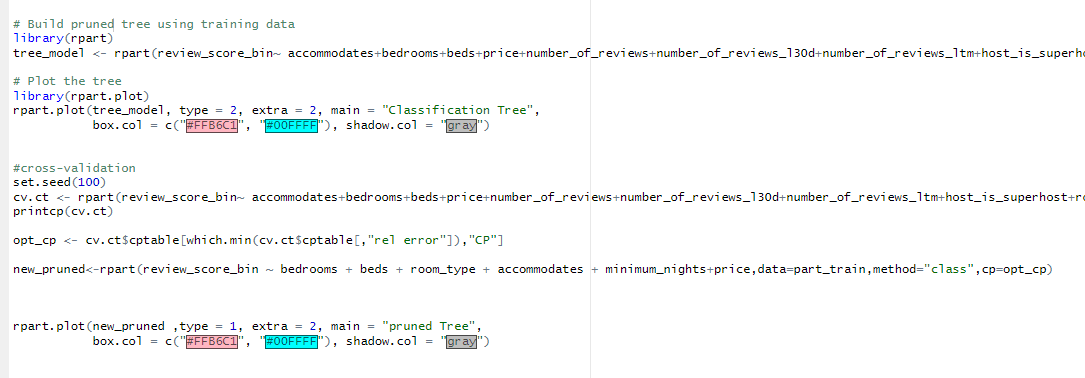
tree\_model <- rpart(review\_score\_bin~ accommodates+bedrooms+beds+price+number\_of\_reviews+number\_of\_reviews\_l30d+number\_of\_reviews\_ltm+host\_is\_superhost+room\_type+minimum\_nights , data = part\_train, method = "class")

****

****

B. Determine the ideal size of your tree using cross-validation.

**Pruned tree**

****

CP nsplit rel error xerror xstd

1 0.027079 0 1.00000 1.0000 0.13476

2 0.021277 11 0.70213 1.2128 0.14568

3 0.014184 15 0.61702 1.4894 0.15741

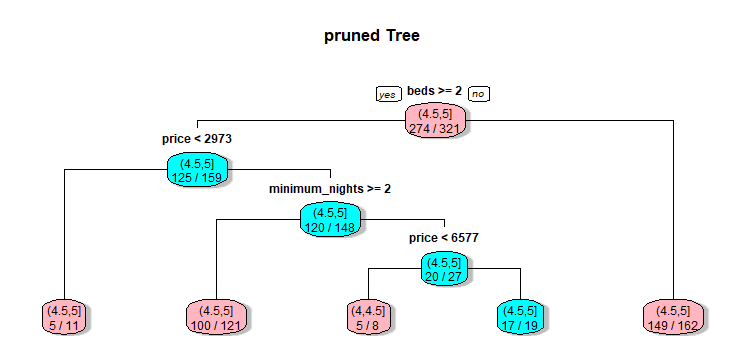
4 0.010638 18 0.57447 1.5745 0.16055

5 0.001000 20 0.55319 1.6383 0.16278

he "rel error" (relative error) column in the output shows the reduction in mean squared error (MSE) achieved relative to the initial MSE at each splitting step of the tree. The "xerror" (cross-validation error) column shows the cross-validation estimate of the MSE for each potential split, which is based on the performance of the model on the validation set during cross-validation.

The purpose of using the relative error over cross-validation error to find the optimal cp is to avoid overfitting to the training data. If we simply choose the split with the lowest cross-validation error, we may end up with a tree that is too complex and overfits the training data, leading to poorperformance on new data. By instead selecting the split with the largest reduction in relative error, we are able to find a balance between model complexity and performance, resulting in a more generalizable model.

**Now from the above analysis we have a pruned tree.**

****