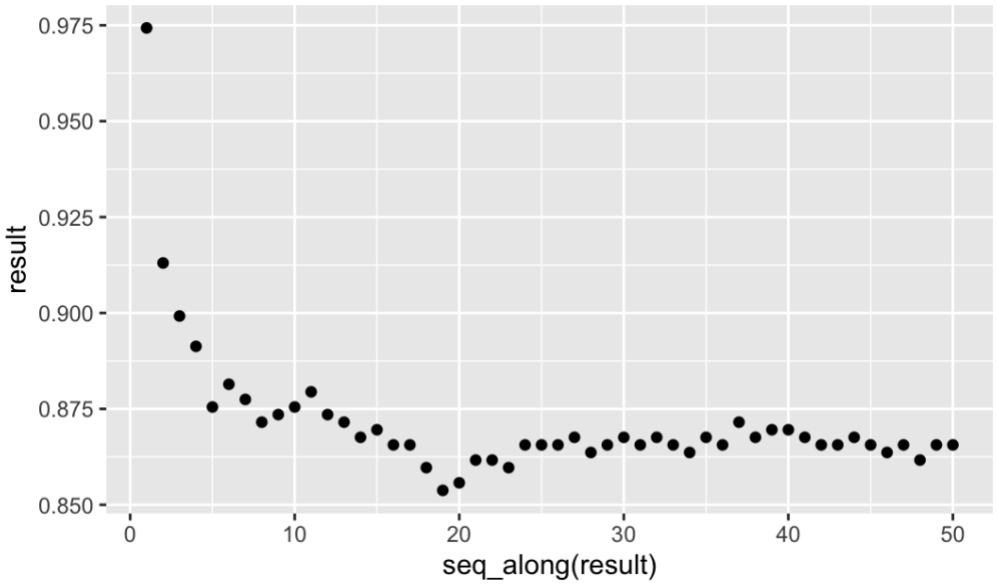
Hands-on Exercise 4

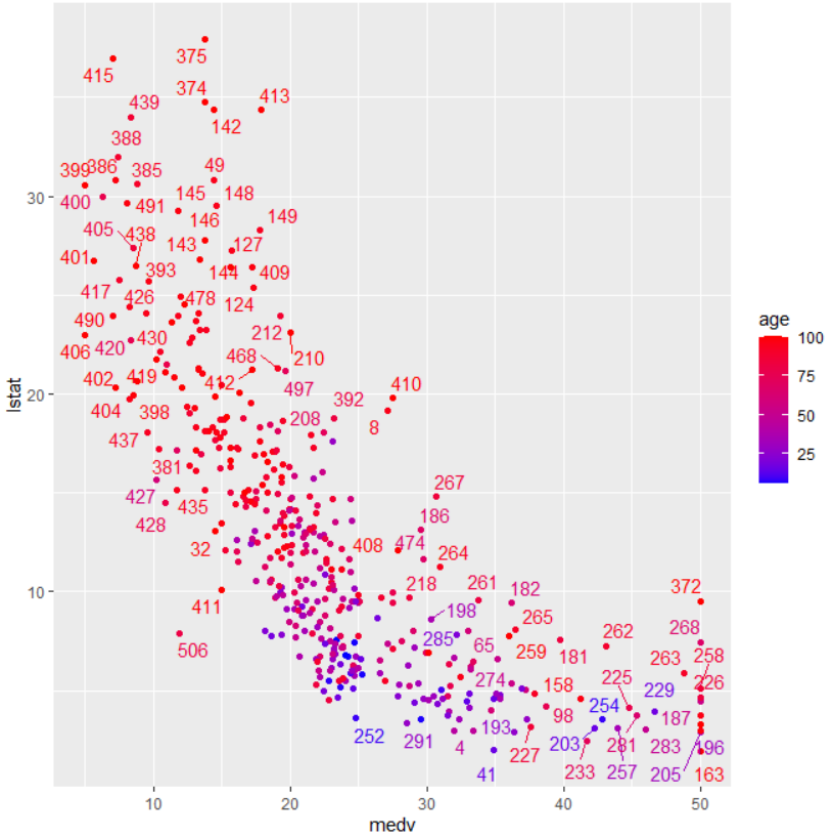
Follow the examples in Ch. 7 and Ch. 13 and the provided videos to build a few knn models and then a few ensemble models of the **Boston housing** dataset. **In hands-on report 2, you have partitioned the dataset. Use the same training and validation (or testing) datasets used in hands-on 2 to train and select (evaluate) the model for this assignment**. Specifically, your report should include the following:

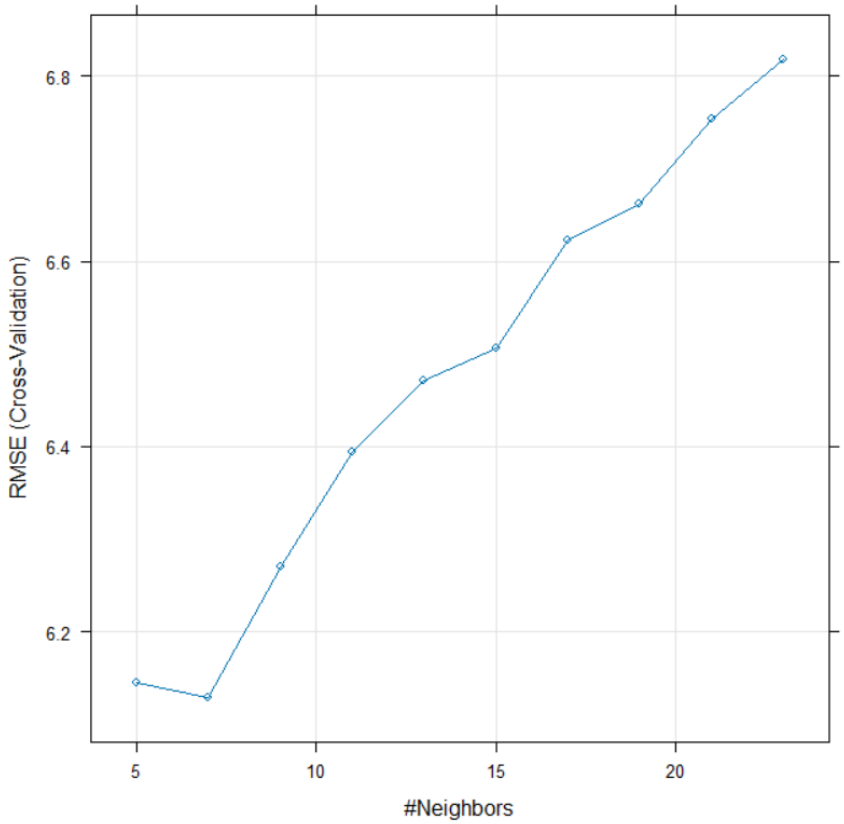
- **How to develop knn models, including details on model setting. You should build a few knn models, analyze each model, justify and select the best knn model candidate.**

The k-nearest-neighbors (k-NN) method is used to identify k records in the training dataset that are similar to a new record that we wish to classify. The k-NN method can be used for classification of a categorical outcome or prediction of a numerical outcome. In order for us to develop a k-Nearest Neighbors model, we have to start by preparing the data through feature scaling. We also have to put into consideration feature selection, because irrelevant features can negatively impact the model's accuracy. After we have completed the data preparation, we have to determine the best number of neighbors, k, we will do that through cross validation, which involves testing different values for k. Through that we select the one that optimizes model performance, usually found by plotting an accuracy curve to identify the "elbow" where the accuracy stabilizes. While setting up the model we have to choose a distance metric like Euclidean or Manhattan, and decide on uniform or distance based weighting, where closer neighbors may carry more influence on the prediction. Training the model involves fitting on a training set and evaluating performance using metrics like accuracy, precision, and recall. Further, hyperparameter tuning can refine the choice of k and weighting through grid or randomized search. This process helps ensure that the k-NN model generalizes well to new data, particularly when combined with cross-validation to validate the chosen parameters.

For our analysis, we used different values of k (5, 7, 9, 11, 13, 15, 17, 19, 21, and 23) using cross-validation in order to select the optimum value of k . We analyzed each model's performance based on Root Mean Squared Error (RMSE). The model where k=7 resulted in a RMSE of 6.129375. We concluded that this was the best model for this dataset compared to the other 8 values of k we analyzed. This model also resulted in an adjusted R squared of 0.5482680.


0.975
0.950
0.925
0.900
0.875
0.850
result
0
10
20
30
40
50
seq_along(result)


lstat
medv
age
100
75
50
25
10
20
30
40
50



k	RMSE	Rsquared	MAE
5	6.144955	0.5514107	4.205510
7	6.129375	0.5482680	4.241578
9	6.270533	0.5245934	4.327374
11	6.393566	0.5053889	4.371832
13	6.471554	0.4941659	4.467942
15	6.505928	0.4893752	4.507972
17	6.622009	0.4709143	4.624317
19	6.662144	0.4653763	4.646187
21	6.754170	0.4522536	4.747529
23	6.817500	0.4429864	4.810979

- **How to develop ensemble models. You should build a few types of ensemble models, such as boosting, bagging, and combining multiple best model candidates developed earlier. Analyze each ensemble model, justify and select the best ensemble model candidate.**

Ensemble refers to the combination of multiple supervised models into a big model. Ensemble improves predictivity, as using the average of multiple predictors can lead to a smaller error variance.

RMSE BOOSTING MODEL:

```
      RMSE  Rsquared       MAE
3.3263643 0.8841947 2.1679069
```

In the boosting model, we achieve a high r-squared value of 0.884 indicating a good fit of the model. In addition, both RMSE and MAE are low, showing good predictive accuracy.

METRICS STACKED MODEL

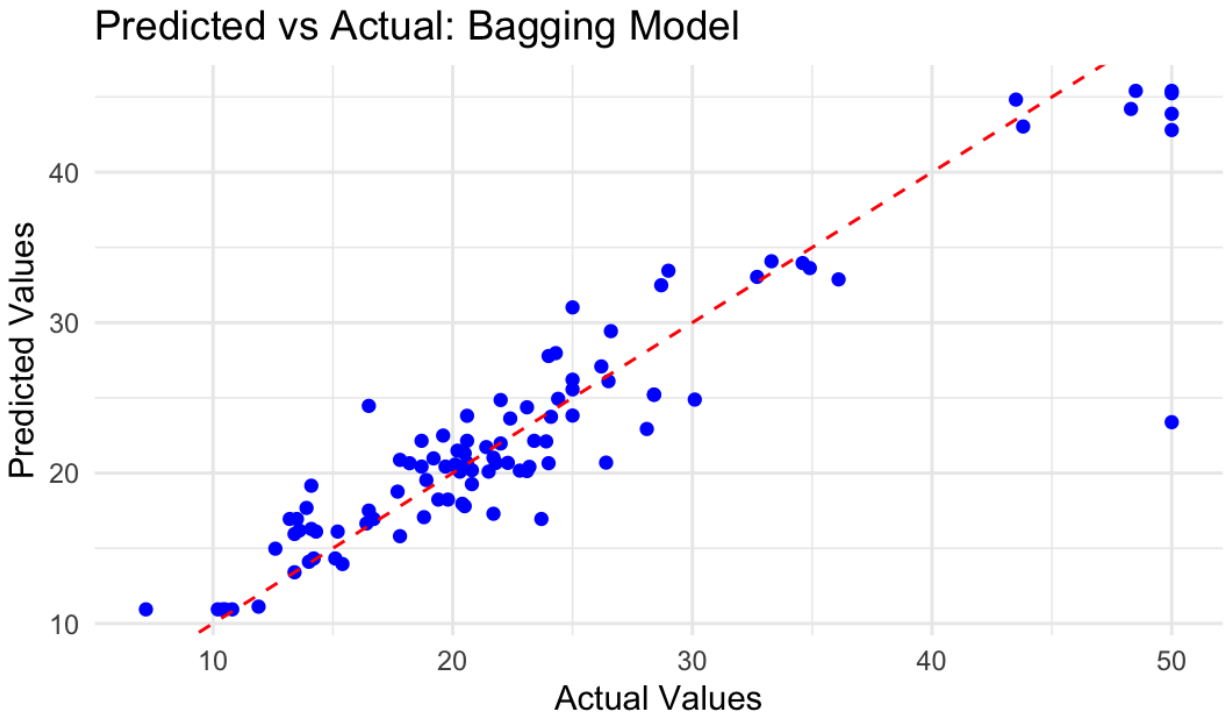
```
The following models were ensembled: bagging, boosting

Model Importance:
 bagging boosting
       0        1

Model accuracy:
   model_name metric    value        sd
       <char> <char>    <num>     <num>
1:   ensemble   RMSE 3.367158 0.2413191
2:    treebag   RMSE 4.283147 1.0709895
3:        gbm   RMSE 3.253577 0.9489162
>
> # Make predictions with the stacked ensemble model
> ensemble_predictions <- predict(ensemble_model, testData)
>
> # Evaluate the performance of the ensemble model
> postResample(ensemble_predictions, testData$medv)
     RMSE  Rsquared       MAE
3.3059119 0.8863216 2.1532431
```

For the stacked model that combines bagging and boosting, the individual metrics do not outperform the stacked model. In terms of individual metrics the bagging model has an RMSE of 4.28 and the boosting model has an RMSE of 3.25. The combined ensemble model has an

RMSE of 3.31 with an r-squared of 0.886 and MAE of 2.15. The stacked model improves upon the bagging model alone with a slightly lower MAE than the boosting model.



In the plot above, the bagging model demonstrates a generally linear relationship between the actual and predicted models. There is notable spread around the line which suggests some variability in the predictions especially at higher levels.

The stacked ensemble model provides the best choice for predictive performance. It has the best overall r-squared (0.886) a slightly lower RMSE (3.31) and MAE (2.15). These values show that the stacked ensemble model is slightly better at predicting values on the dataset. In addition, by combining bagging and boosting, we leverage the strength of both models while reducing overfitting.

- **Compare and evaluate all of the prediction models developed for this dataset (i.e., hands-on 2, 3, 4). Discuss what would be the recommended model and its performance. Interpret the results of the model, make recommendations/suggestions.**

At this point, we have created and developed useful models that have allowed us to understand our data better, capture patterns or relationships, look at predictive performance model results, and identify important tradeoffs. By running these various models, we are able to advance our

analysis, identify strong models, blend models, and make appropriate recommendations. Below is a comparison and evaluation of the models developed.

Multiple Linear Regression Models (MLR)

Out of all multiple linear regression models, we figured out that the models with the "backward" and “both directions” selection are the two most robust because they have better predictive performance overall. Based on adjusted r-squared metric and by striving to reduce the risk of excluding important variables in a simple manner helped us choose a single model (model with "backward" selection has an adjusted r square of 0.8459 and RMSE of 3.897 rounded) in hands on exercise 2.This backward selection regression model is the best model that also has the benefit of interpretability in a linear fashion. We can drill down into each predictor and assess specific impact.

Regression Trees

When running regression tree models, we figured out that model 3 had the lowest error values (MSE of 24.261 rounded or 4.93 RMSE) and is the best out of all regression trees to capture non-linear relationships. This type of model can also capture nonlinear relationships but we may have to make some trade offs like ease of interpretation, significant computational resource usage, and or overfitting.

Neural Network

When running neural network models, we figured out that the 8 neuron model has the highest performance accuracy values (lowest RMSE of 3.65) yet the 5 neuron model provides a nice balance between simplicity and robustness (RMSE of 3.91). When running tree models we may have to watch out for potential favoring of predictors but this model can capture nonlinear relationships and not worry so much about outliers.

KNN Model

From the scatter plot, with LSTAT on the y-axis and MEDV on the x-axis, there is a nonlinear relationship among the proportion of lower status of the population (LSTAT) and median value of owner-occupied homes in $1000s (MEDV). Also there is a negative relationship among these variables and a higher proportion of lower-status tends to have lower median home values. Interestingly, there are no more records that are above 50 which might mean a limit or other reason for the cutoff.

From the plot displaying RMSE (cross-validation) on the y-axis and #Neighbors on the x-axis, it looks like error decreases after 5 and increases after 7. When running 10 different k values, we figured out that the lowest RMSE value is 6.1294 or when k = 7, the lowest R squared value of 0.4430 is k=23 and MAE is lowest of 4.2055 or when k = 5. It looks like the first two or when k=5 or 7 will bring the strongest predictive performance. With KNN models, we can take advantage of its simplicity and lack of parametric assumptions but we may have to watch out for the required records needed for training as the number of predictors increases or other relevant difficulties.

Ensembles

Speaking of predictive performance, we've been using RMSE to help us identify which models make less predictive errors. When using ensembles (boosting), we are able to pinpoint that the RMSE = 3.3264. When using ensembles (bagging and boosting) the RMSE values increase and for “ensemble” its 3.3671, for “treebag” its 4.2831, and for “GBM” or gradient boosting machine its 3.2536. Finally, the combined ensemble (stacked) model has an RMSE of 3.3059 with an r-squared of 0.886 and MAE of 2.15. Out of all ensembles, it looks like the GBM ensemble model has the lowest RMSE value of 3.2536. Relevantly, the plot that looks at predicted and actual values from the bagging model, shows that there is not a lot of deviation from the reference line and that the model has good predictive performance.

With these models, we can uncover different types of relationships, gain a better understanding of our data, balance strengths and weaknesses of each model in our decision making. Similarly in the previous hands on exercise, we focused on RMSE to identify which model was better at making predictions. From this exercise, we can tell that the best models with predictive strength are ensembles and the GBM model has the lowest RMSE value of 3.25 while the combined (stacked) ensemble model has an RMSE of 3.31. Other useful models include backward selection MLR model because it is easy to interpret and it has a RMSE of 3.90. The 5-neuron neural network has a RMSE of 3.91 and is a simpler robust alternative (regarding neural networks) for capturing those non-linear patterns. Model 3 from the regression tree is also recommended because it can handle nonlinear or complex relationships despite a slightly higher RMSE (4.93 RMSE). For KNN models, we have to watch for the high RMSE values but these models can be helpful when looking at local patterns or effects. In this exercise, we discovered that running (ensemble) models, that capture nonlinear effects, can return a lower RMSE value in our data. Overall, we expanded our analysis to obtain a comprehensive understanding of the data, compared models, assess relevant tradeoffs, and maintained a focus on predictive strength.