# Tree-based models

Brooke Anderson

March 22, 2016

# Tuning

```r
rmsle_fun <- function(data, lev = NULL,
                      model = NULL, ...){
  log_p_1 <- log(data$pred + 1)
  log_a_1 <- log(data$obs + 1)
  sle <- (log_p_1 - log_a_1)^2
  rmsle <- sqrt(mean(sle))
  names(rmsle) <- "rmsle"
  return(rmsle)
}

fitControl <- trainControl(method = "cv",
                           number = 5,
                           summaryFunction = rmsle_fun)
```

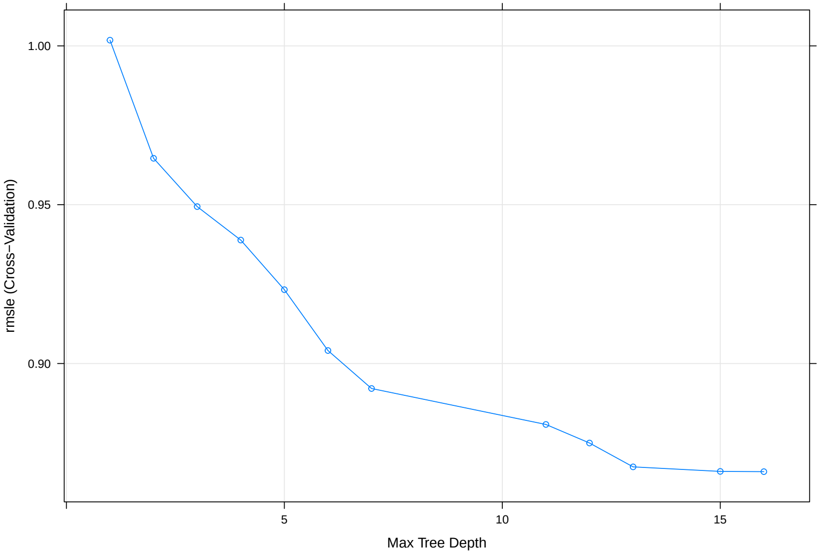
# Regression tree

| caret method | package(s) | tuning parameters |
|---|---|---|
| ctree | party | mincriterion |
| ctree2 | party | maxdepth |
| evtree | evtree | alpha |
| rpart | rpart | cp |
| rpart1SE | rpart | None |
| rpart2 | rpart | maxdepth |
| M5 | RWeka | pruned, smoothed, rules |

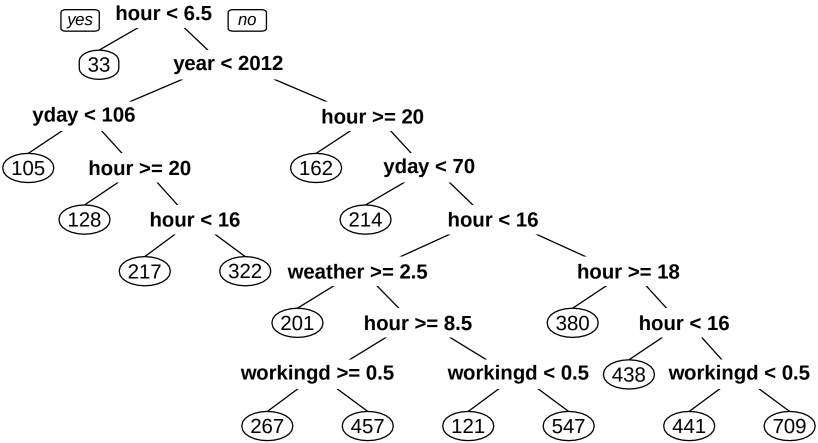
# Regression tree model

```
set.seed(825)
mod_1 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour +
                 month + yday, data = train,
               method = "rpart2",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneLength = 12)
```

# Regression tree model



# Regression tree model



# Regression tree model

Check how the model did in the training data:

```r
train_preds <- predict(mod_1, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 0.8572327
```

# Regression tree model

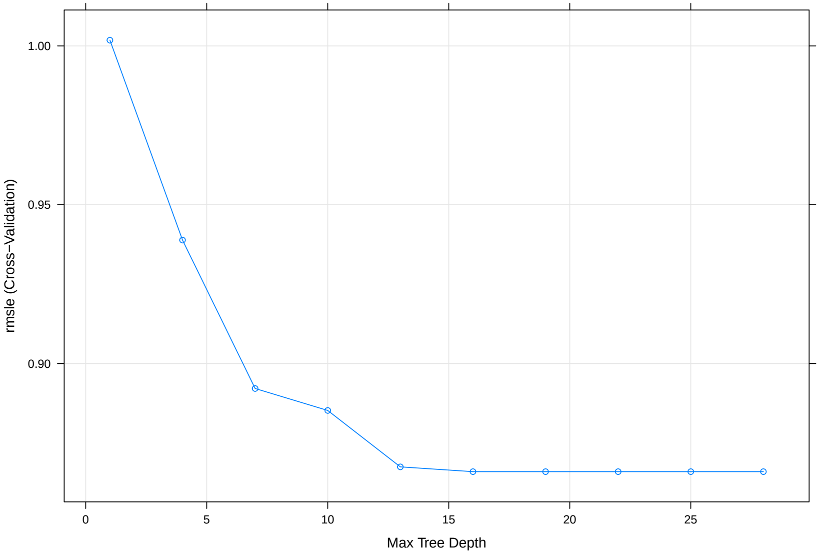
I decided to try to look at larger trees. `rpart2` optimizes on `maxdepth`. Here's what the help file for `rpart.control` says about that parameter:

> *“`maxdepth`: Set the maximum depth of any node of the final tree, with the root node counted as depth 0. Values greater than 30 rpart will give nonsense results on 32-bit machines.”*

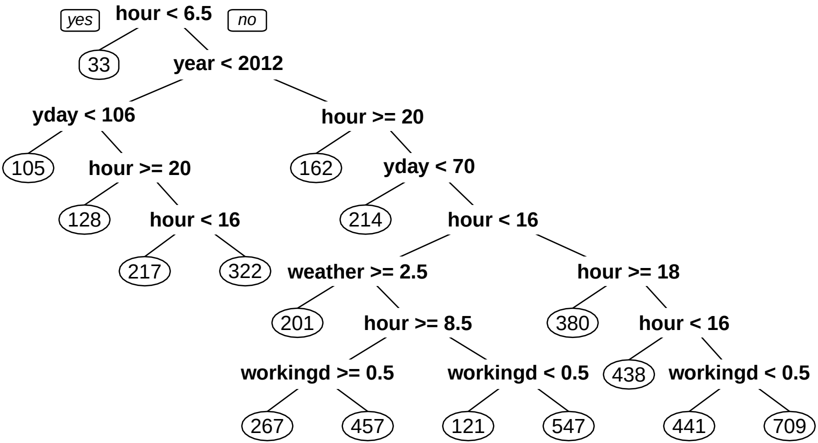
# Regression tree model

```r
set.seed(825)
mod_2 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour +
                 month + yday, data = train,
               method = "rpart2",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneGrid = data.frame(maxdepth =
                              seq(from = 1,
                                  to = 30,
                                  by = 3)))
```

# Regression tree model



# Regression tree model



# Regression tree model

Check how the model did in the training data:

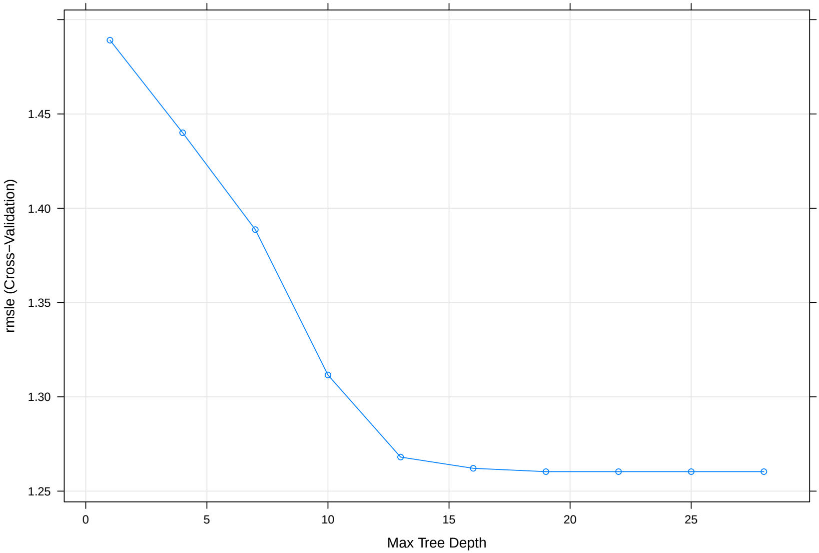
```r
train_preds <- predict(mod_2, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 0.8572327
```

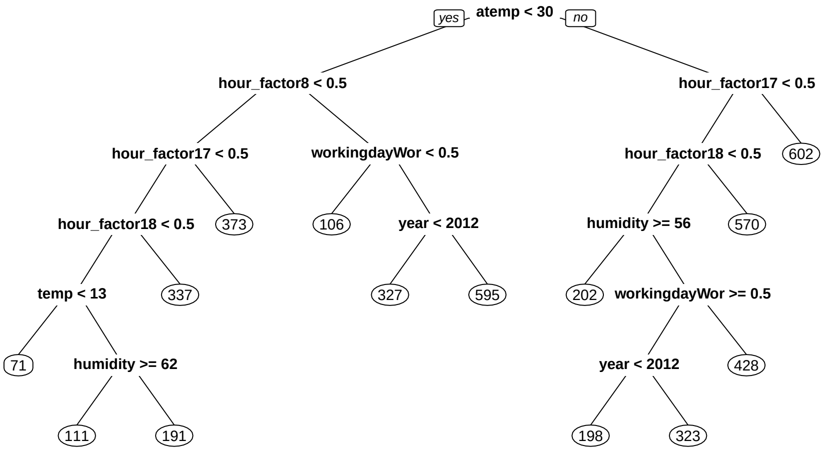
# Regression tree model

```r
set.seed(825)
mod_3 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour_factor +
                 month + yday, data = train,
               method = "rpart2",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneGrid = data.frame(maxdepth =
                              seq(from = 1,
                                  to = 30,
                                  by = 3)))
```

# Regression tree model



# Regression tree model



# Regression tree model

Check how the model did in the training data:

```
train_preds <- predict(mod_3, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 1.290799
```

# Conditional regression tree model

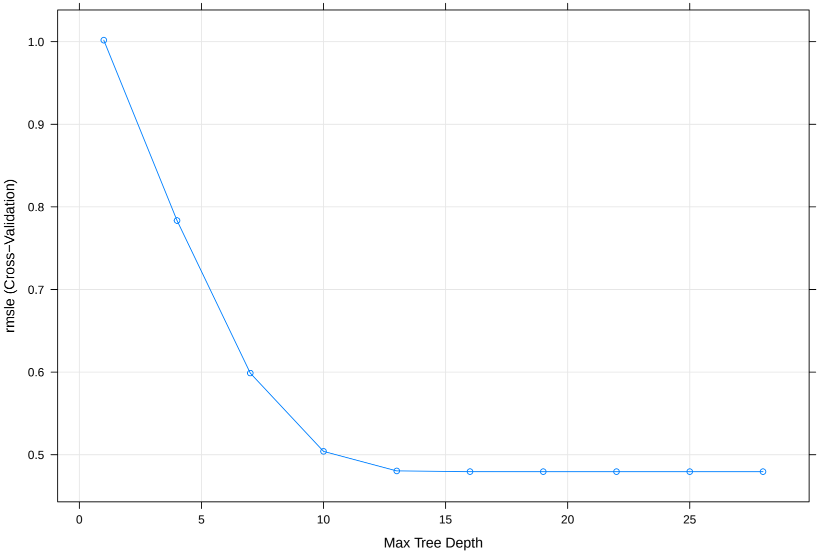
From the `ctree` vignette:

> *"We present a unified framework embedding recursive binary partitioning into the well defined theory of permutation tests developed by Strasser and Weber (1999). The conditional distribution of statistics measuring the association between responses and covariates is the basis for an unbiased selection among covariates measured at different scales. Moreover, multiple test procedures are applied to determine whether no significant association between any of the covariates and the response can be stated and the recursion needs to stop."*

# Conditional regression tree model

```r
set.seed(825)
mod_4 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour +
                 month + yday, data = train,
               method = "ctree2",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneGrid = data.frame(maxdepth =
                              seq(from = 1,
                                  to = 30,
                                  by = 3)))
```

# Conditional regression tree model



# Conditional regression tree model

Check how the model did in the training data:

```r
train_preds <- predict(mod_4, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 0.3667714
```

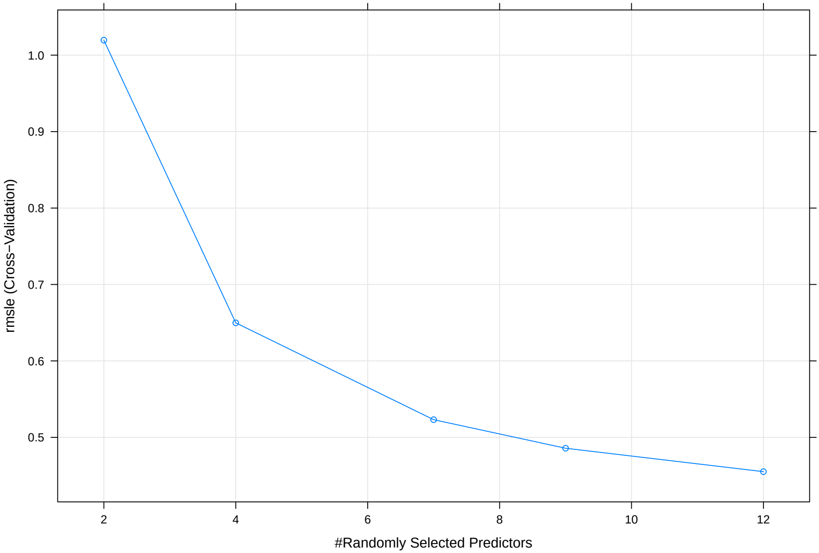
# Random forest

| caret method | package(s) | tuning parameters |
|---|---|---|
| cforest | party | mtry |
| extraTrees | extraTrees | mtry, numRandomCuts |
| parRF | e1071, randomForest, foreach | mtry |
| ranger | e1071, ranger | mtry |
| rf | randomForest | mtry |
| rfRules | randomForest, inTrees, plyr | mtry, maxdepth |
| RRF | randomForest, RRF | mtry, coefReg, coefImp |
| RRFglobal | RRF | mtry, coefReg |
| qrf | quantregForest | mtry |

# Conditional random forest

```r
set.seed(825)
mod_5 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour +
                 month + yday, data = train,
               method = "cforest",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneLength = 5,
               controls = cforest_unbiased(ntree = 50))
```

# Conditional random forest



# Conditional random forest

Check how the model did in the training data:

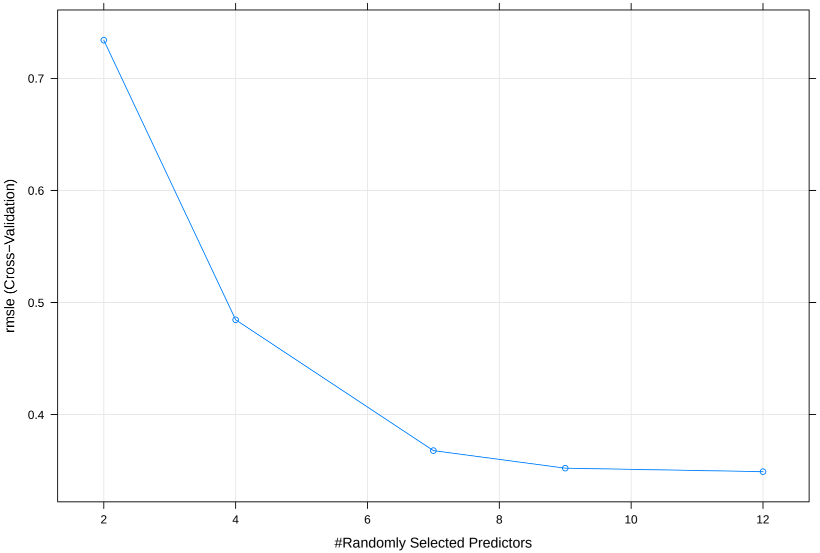
```r
train_preds <- predict(mod_5, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 0.3849723
```

# Random forest

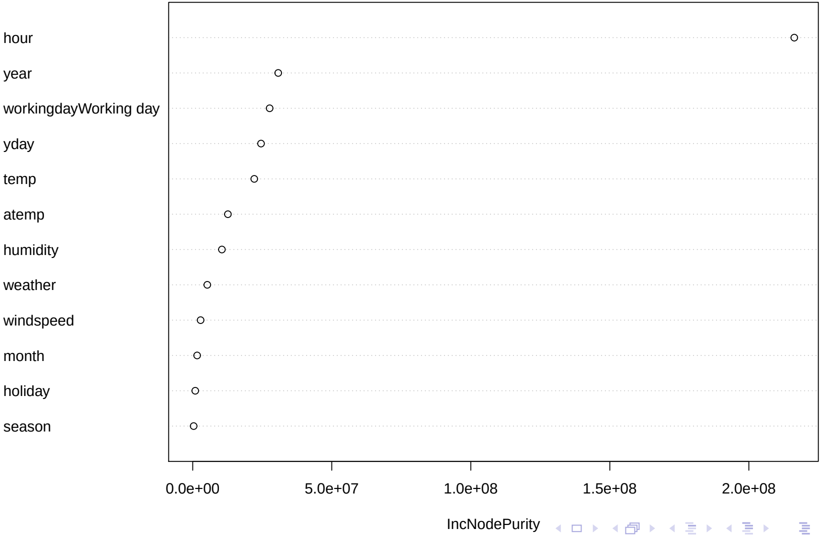
```r
set.seed(825)
mod_6 <- train(count ~ season + holiday +
                 workingday + weather +
                 temp + atemp + humidity +
                 windspeed + year + hour +
                 month + yday, data = train,
               method = "rf",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               tuneLength = 5,
               ntree = 50)
```

# Random forest



# Random forest

```
varImpPlot(mod_6$finalModel, type=2,
           main = "")
```



# Random forest

Check how the model did in the training data:

```r
train_preds <- predict(mod_6, newdata = train)
rmsle(train_preds, train$count)
```

```
## [1] 0.1804334
```

# Boosting

| caret method | package(s) | tuning parameters |
|---|---|---|
| blackboost | party, mboost, plyr | mstop, maxdepth |
| bstTree | bst, plyr | mstop, maxdepth, nu |
| gbm | gbm, plyr | n.trees, interaction.depth, |
| .. | .. | shrinkage, n.minobsinnode |