STAT 388 Project

Tina Yang

Charles Hwang

Professor Matthews

STAT 388-001

10 December 2019

(1) Introduction: We are building a model to predict the meter readings of energy usage in buildings in a series of given data sets. We are doing this as part of the ASHRAE Great Energy Predictor III competition on Kaggle. There are four types of energy being used: chilled water, electric, hot water, and steam. The data consist of observations from over 1,000 buildings over a three-year timeframe. There are designated training and test data sets with the columns "building_id", energy type ("meter"), and "timestamp" containing $20,216,100$ and $41,697,600$ observations, respectively. There are also data sets containing weather information at the time of the meter reading for each (with columns like "air_temperature", "cloud_coverage", "dew_temperature", "sea_level_pressure", "wind_direction", and "wind_speed") and a "building_metadata" data set (with columns like "primary_use", "square_feet", "year_built", and "floor_count").

(2) Methods: First, we read the data into RStudio (this took around 20 minutes due to the size of the files) and merged the appropriate data sets into the training or test data set using the join() function from the "dplyr" package. Then we looked at different variables and found that a logarithmic transformation on the "meter_reading" variable in the training data set would be appropriate. We built a classification and regression tree (CART) using the rpart() function from the "rpart" package and obtained its predicted values and cross-validation error. We pruned the tree with the prune() function but found no significant difference in predicted values. Finally, the predictions were copied to a newly-created data set "tpdf", untransformed, and written to a CSV file to be uploaded to the Kaggle competition webpage. All of the code for this project can be found in the (5) Appendix of this report.

(3) Results: The tree we fit had most of its splits on the "square_feet" and "meter" variables, with one on the "air_temperature" variable. The results can be found in the (5.2) Tree subsection of this report. After uploading the CSV file to the Kaggle competition webpage, our calculated Kaggle score was 1.850. The Kaggle score is calculated as the Root Mean Squared Logarithmic Error (RMSLE) of the model, which is given by

$$\epsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\ln(p_i+1) - \ln(a_i+1))^2}\,,$$

where ϵ is the RMSLE value (score), n is the total number of observations in the (public/private) data set, p_i is our prediction of target, and a_i is the actual target for i. Because the score is based on the error of the model, a lower score indicates a model that is more well-fit to the data provided.

(4) Conclusions/Future Work: We conclude that our Kaggle score was 1.850. The Kaggle competition awards cash prizes to the teams with the five lowest Kaggle scores and (at the time of the creation of this report) had 4,071 competitors in total, many of which make numerous submissions over the course of the competition to improve upon their score and have several advanced resources and programs dedicated to prediction outside of R to use that we do not have. In light of this information, this score is good. There are many different things we would do in a similar project in the future if we had more time, including: * Fitting a larger tree or multiple trees on the different factors of a variable ("meter", for example) * Fitting a tree or trees with functions from different packages or a different set.seed() value + We experimented with fitting a series of

trees using the tree() function from the "tree" package; however, the resulting Kaggle score of the model was greater than the one that resulted from the model in this report. * Using different tree-based methods like bagging * Fitting a random forest + When attempting to fit a random forest, RStudio yielded an error stating "vector memory exhausted (limit reached?)". Thus, fitting a random forest on this data would likely require an adjustment to the memory settings of the computer that the code is being run on or running the code on a computer with more RAM. * Experimenting with other types of prediction models like gradient boosted models (GBMs) to see how the data would be handled

(5) Appendix

(5.1) Data Cleaning

```
rm(list=ls())
building_metadata <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive Ana
# sample_submission <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive
test <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive Analytics/ashra
train <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive Analytics/ashra
weather_test <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive Analyti
weather_train <- read.csv(file="/Users/newuser/Desktop/Notes/Undergraduate/STAT 338 - Predictive Analyti
library(car)
library(dplyr)
library(rpart)
train <- left_join(train,building_metadata,by="building_id")
train <- left_join(train,weather_train,by=c("site_id","timestamp"))
test <- left_join(test,building_metadata,by="building_id")
test <- left_join(test,weather_test,by=c("site_id","timestamp"))
summary(train$meter_reading)
```

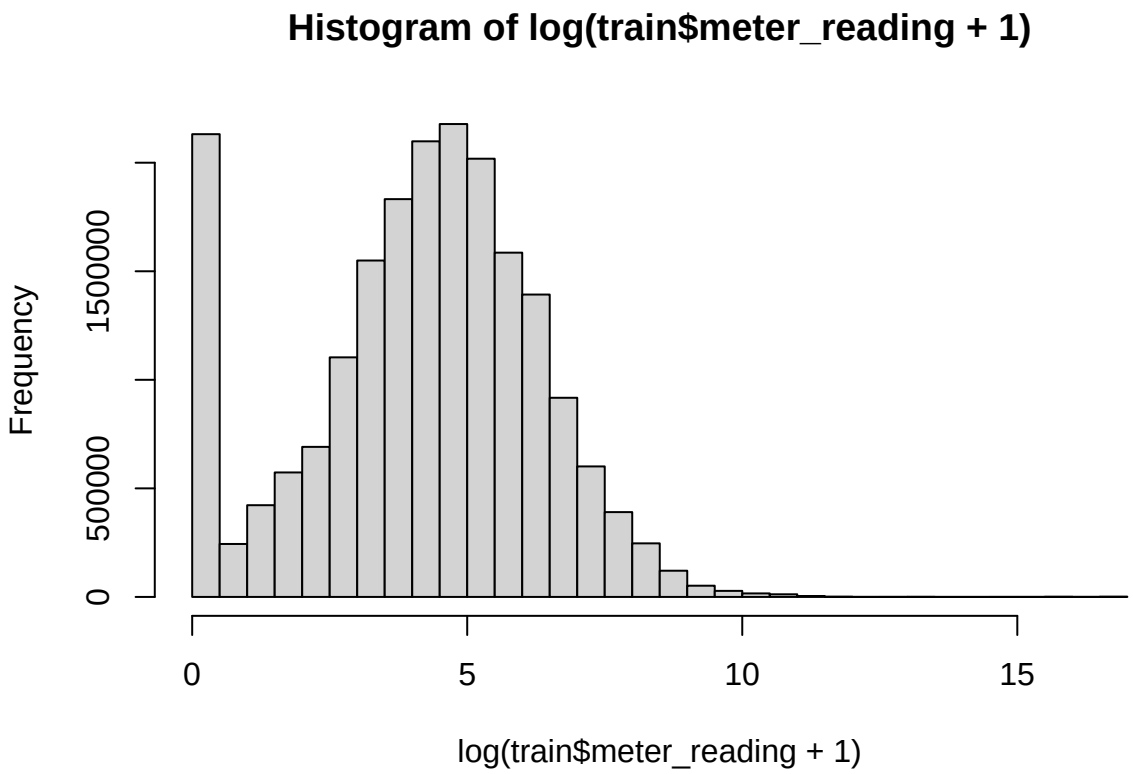
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##        0       18       79     2117      268 21904700
```

```
hist(train$meter_reading)
```


```
summary(log(train$meter_reading+1)) # Looking at summary of possible logarithmic transformation
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.960   4.379   4.146   5.595  16.902
```

```
hist(log(train$meter_reading+1)) # Performing a logarithmic transformation on meter reading
```



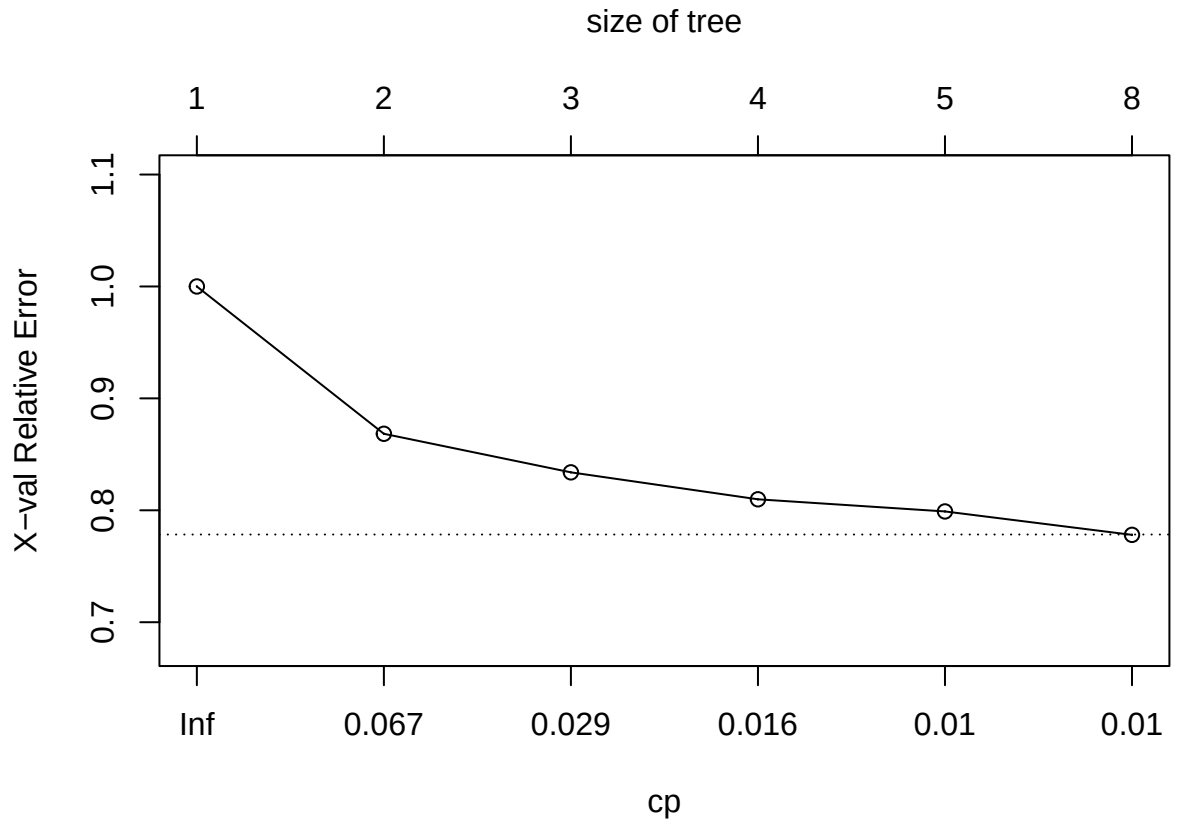

```
train$meter_reading <- log(train$meter_reading+1) # Done
train$timestamp <- as.integer(train$timestamp)
test$timestamp <- as.integer(test$timestamp)
```

(5.2) Tree

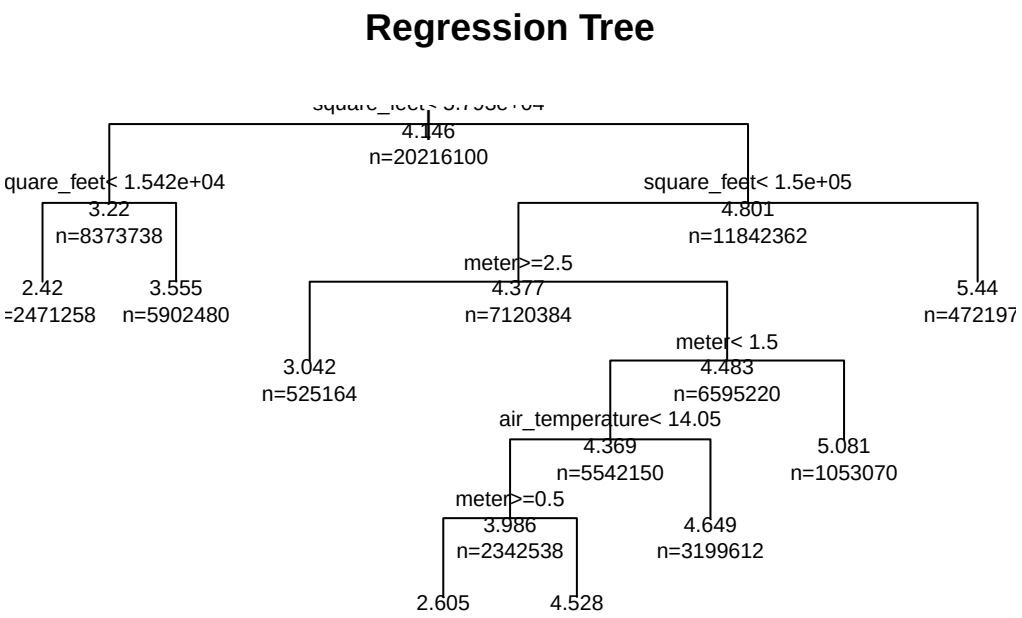
```
set.seed(1012)
tree <- rpart(meter_reading~.,data=train)
printcp(tree)
```

```
##
## Regression tree:
## rpart(formula = meter_reading ~ ., data = train)
##
## Variables actually used in tree construction:
## [1] air_temperature meter           square_feet
##
## Root node error: 93120618/20216100 = 4.6063
##
## n= 20216100
##
##         CP nsplit rel error  xerror       xstd
## 1 0.131628      0   1.00000 1.00000 0.00030318
## 2 0.034468      1   0.86837 0.86837 0.00030989
## 3 0.024086      2   0.83390 0.83390 0.00030822
## 4 0.010850      3   0.80982 0.80982 0.00031038
## 5 0.010012      4   0.79897 0.79897 0.00030840
## 6 0.010000      7   0.76893 0.77805 0.00030622
```

```
plotcp(tree)
```




```
plot(tree,uniform=TRUE,main="Regression Tree")
text(tree,use.n=TRUE,all=TRUE,cex=.7)
```




```
summary(tree)
```

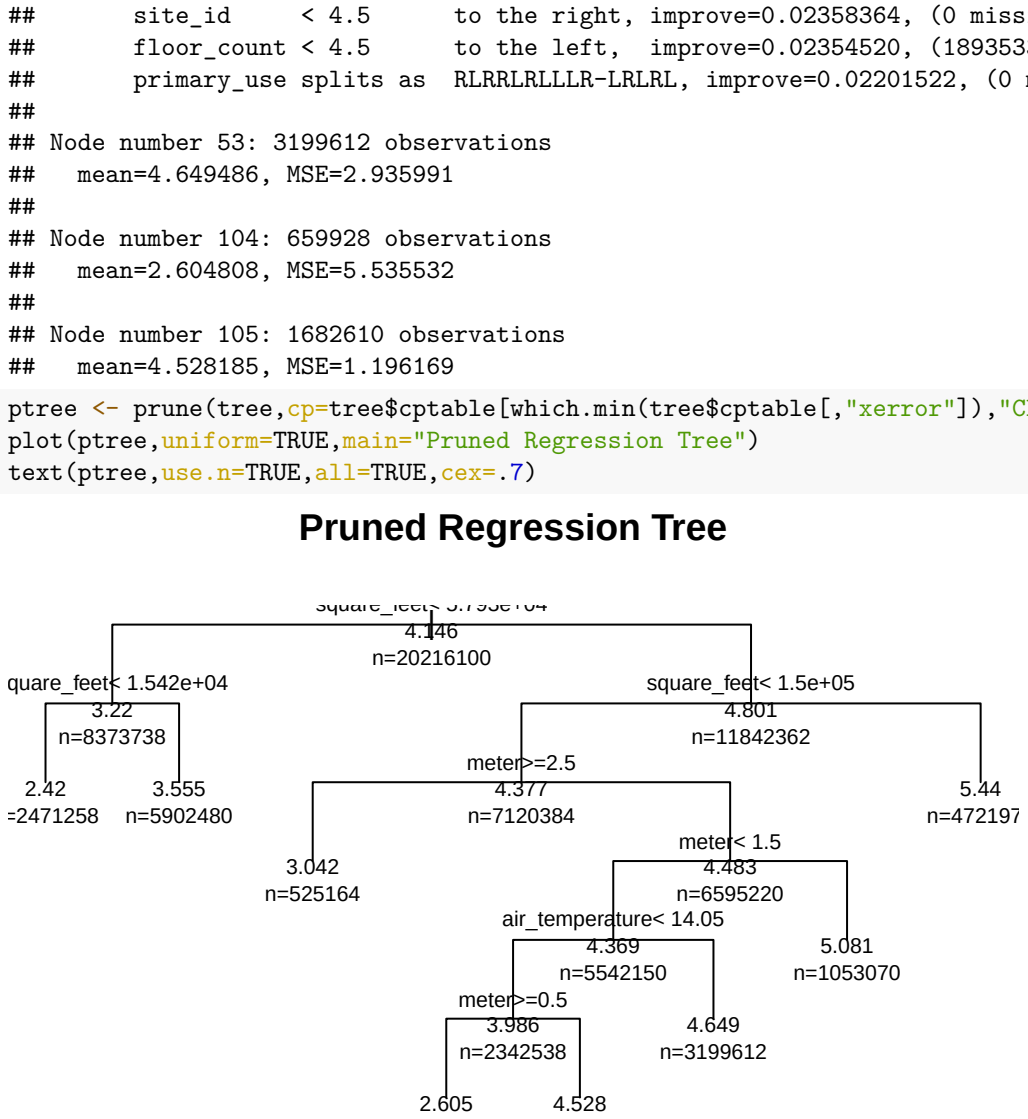
```
## Call:
## rpart(formula = meter_reading ~ ., data = train)
```

```
##   n= 20216100
##
##           CP nsplit rel error    xerror         xstd
## 1 0.13162845      0 1.0000000 1.0000001 0.0003031817
## 2 0.03446760      1 0.8683716 0.8683717 0.0003098897
## 3 0.02408583      2 0.8339040 0.8339043 0.0003082221
## 4 0.01084998      3 0.8098181 0.8098185 0.0003103810
## 5 0.01001184      4 0.7989681 0.7989686 0.0003083959
## 6 0.01000000      7 0.7689326 0.7780499 0.0003062187
##
## Variable importance
##     square_feet           meter     primary_use     building_id air_temperature
##              71              13               6               4               2
##         site_id dew_temperature
##               2               1
##
## Node number 1: 20216100 observations,    complexity param=0.1316284
##   mean=4.145766, MSE=4.60626
##   left son=2 (8373738 obs) right son=3 (11842362 obs)
##   Primary splits:
##       square_feet < 57926    to the left,  improve=0.13162840, (0 missing)
##       floor_count < 4.5      to the left,  improve=0.02851350, (16709167 missing)
##       building_id < 872.5    to the left,  improve=0.02735902, (0 missing)
##       site_id     < 8.5      to the left,  improve=0.02722985, (0 missing)
##       primary_use splits as  RLRRLRRLLLLLRLRL, improve=0.01275902, (0 missing)
##   Surrogate splits:
##       primary_use splits as  RLLRRLRLRLLLRRLL, agree=0.628, adj=0.101, (0 split)
##       building_id < 763.5    to the left,  agree=0.612, adj=0.062, (0 split)
##       site_id     < 5.5      to the left,  agree=0.598, adj=0.030, (0 split)
##       meter       < 0.5      to the left,  agree=0.586, adj=0.001, (0 split)
##
## Node number 2: 8373738 observations,    complexity param=0.02408583
##   mean=3.219771, MSE=2.978581
##   left son=4 (2471258 obs) right son=5 (5902480 obs)
##   Primary splits:
##       square_feet < 15417    to the left,  improve=0.08992465, (0 missing)
##       building_id < 1137.5   to the left,  improve=0.02918354, (0 missing)
##       site_id     < 13.5     to the left,  improve=0.02426434, (0 missing)
##       primary_use splits as  LLRRLRLLLLLLRRRL, improve=0.02203266, (0 missing)
##       floor_count < 4.5      to the left,  improve=0.02027522, (6438612 missing)
##   Surrogate splits:
##       primary_use splits as  RRRRRLRLLLLRRRRR, agree=0.721, adj=0.054, (0 split)
##       building_id < 2.5      to the left,  agree=0.708, adj=0.011, (0 split)
##
## Node number 3: 11842362 observations,    complexity param=0.0344676
##   mean=4.800537, MSE=4.722152
##   left son=6 (7120384 obs) right son=7 (4721978 obs)
##   Primary splits:
##       square_feet < 150028   to the left,  improve=0.057395590, (0 missing)
##       meter       < 2.5      to the right, improve=0.025650490, (0 missing)
##       building_id < 1146.5   to the left,  improve=0.009715203, (0 missing)
##       primary_use splits as  RRRRRRRLLR-LRLRL, improve=0.008817663, (0 missing)
##       site_id     < 12.5     to the left,  improve=0.007652370, (0 missing)
##   Surrogate splits:
```

```
##       primary_use splits as  LRLRLLLLRL-LRLLL, agree=0.618, adj=0.043, (0 split)
##
## Node number 4: 2471258 observations
##   mean=2.419932, MSE=2.08927
##
## Node number 5: 5902480 observations
##   mean=3.554648, MSE=2.970928
##
## Node number 6: 7120384 observations,    complexity param=0.01084998
##   mean=4.376582, MSE=4.006724
##   left son=12 (525164 obs) right son=13 (6595220 obs)
##   Primary splits:
##       meter       < 2.5      to the right, improve=0.03541457, (0 missing)
##       building_id < 1222     to the left,  improve=0.01581314, (0 missing)
##       site_id     < 13.5     to the left,  improve=0.01581314, (0 missing)
##       square_feet < 97230.5  to the left,  improve=0.01403901, (0 missing)
##       floor_count < 6.5      to the left,  improve=0.01027459, (6189622 missing)
##
## Node number 7: 4721978 observations
##   mean=5.439829, MSE=5.12124
##
## Node number 12: 525164 observations
##   mean=3.041669, MSE=6.625775
##
## Node number 13: 6595220 observations,    complexity param=0.01001184
##   mean=4.482879, MSE=3.644979
##   left son=26 (5542150 obs) right son=27 (1053070 obs)
##   Primary splits:
##       meter       < 1.5      to the left,  improve=0.018662130, (0 missing)
##       square_feet < 104144.5 to the left,  improve=0.016898860, (0 missing)
##       floor_count < 4.5      to the left,  improve=0.010889910, (5751678 missing)
##       building_id < 1222     to the left,  improve=0.009972105, (0 missing)
##       site_id     < 13.5     to the left,  improve=0.009972105, (0 missing)
##
## Node number 26: 5542150 observations,    complexity param=0.01001184
##   mean=4.36919, MSE=3.141032
##   left son=52 (2342538 obs) right son=53 (3199612 obs)
##   Primary splits:
##       air_temperature < 14.05    to the left,  improve=0.03405299, (24920 missing)
##       dew_temperature < 12.05    to the left,  improve=0.03041970, (25727 missing)
##       square_feet     < 104144.5 to the left,  improve=0.01894045, (0 missing)
##       meter           < 0.5      to the right, improve=0.01831913, (0 missing)
##       floor_count     < 4.5      to the left,  improve=0.01633656, (4724569 missing)
##   Surrogate splits:
##       dew_temperature    < 9.35     to the left,  agree=0.792, adj=0.506, (0 split)
##       sea_level_pressure < 1020.55  to the right, agree=0.646, adj=0.159, (0 split)
##       building_id        < 997.5    to the right, agree=0.643, adj=0.152, (24920 split)
##       site_id            < 9.5      to the right, agree=0.643, adj=0.152, (0 split)
##       wind_speed         < 6.85     to the right, agree=0.587, adj=0.019, (0 split)
##
## Node number 27: 1053070 observations
##   mean=5.081206, MSE=5.871157
##
## Node number 52: 2342538 observations,    complexity param=0.01001184
```

```
##   mean=3.986341, MSE=3.167208
##   left son=104 (659928 obs) right son=105 (1682610 obs)
##   Primary splits:
##       meter       < 0.5      to the right, improve=0.23635190, (0 missing)
##       building_id < 677      to the right, improve=0.02442087, (0 missing)
##       site_id     < 4.5      to the right, improve=0.02358364, (0 missing)
##       floor_count < 4.5      to the left,  improve=0.02354520, (1893533 missing)
##       primary_use splits as  RLRRLRLLLR-LRLRL, improve=0.02201522, (0 missing)
##
## Node number 53: 3199612 observations
##   mean=4.649486, MSE=2.935991
##
## Node number 104: 659928 observations
##   mean=2.604808, MSE=5.535532
##
## Node number 105: 1682610 observations
##   mean=4.528185, MSE=1.196169
```

```
ptree <- prune(tree,cp=tree$cptable[which.min(tree$cptable[,"xerror"]),"CP"])
plot(ptree,uniform=TRUE,main="Pruned Regression Tree")
text(ptree,use.n=TRUE,all=TRUE,cex=.7)
```




```
cat("Cross validation error:",min(tree$cptable[,"xerror"]))
```

```
## Cross validation error: 0.7780499
```

(5.3) Predictions

The output in this subsection is suppressed because RStudio would not allow it to be knit. Similarly to when we attempted to fit a random forest, as discussed in the fourth bullet point of (4) Conclusions/Future Work, RStudio yielded an error stating "vector memory exhausted (limit reached?)" when this subsection was in the process of being knitted. However, the code will still run without any errors.

```
treepred <- predict(tree,test)
table(treepred)
tpdf <- as.data.frame(rep(0,length(treepred))) # Exporting predictions to CSV file for Kaggle submissio
```

```
tpdf$row_id <- test$row_id
tpdf$meter_reading <- exp(treepred) # Untransforming predictions
tpdf$`rep(0, length(treepred))` <- NULL
write.csv(as.data.frame(tpdf),"STAT 388 Project Submission.csv",row.names=FALSE)
```

(5.4) Score

```
# Kaggle score: 1.850 (88.87%)
1.850 < 2.250
```

```
## [1] TRUE
```

This score, as previously discussed in (4) Conclusions/Future Work, is relatively good, given the resources that other competitors have and the shorter time period we had to complete this project.