Homework 8

Daria Barbour-Brown | Bailey Ho | Warren Kennedy

2023-05-28

Conceptual Problem

Question 1

Explain, in your own words, how a random forest makes predictions.

A random forest uses trees as building blocks to construct more powerful prediction models. This method is an adaptation of Bagging, i.e bootstrap aggregation, where we generate multiple bootstrapped training datasets, train a model on each set, and average all the predictions to obtain our bagging estimator. Random forests provide an improvement over bagged trees by randomizing the sample of predictors that can be used for splitting trees at each split of every bagged tree. This adaption decorrelates the bagged trees from each other increasing the variation accounted for by each tree, thereby making the average of the resulting trees less variable and more reliable. Our final random forest bagging estimator will be used for prediction.

Application Problems

Question 2

Using a linear model, predict bikers with the other variables in Bikeshare (excluding casual and registered). Make sure to use the variables in their appropriate form (i.e. numeric or factors as relevant to the context of the problem.) Perform variable selection as needed to get a strong predictive model. Use training/test splits of your choice and report a RMSE that describes your model's performance.

```r
library(ISLR2)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:ISLR2':
##
##     Boston
```

```r
data <- Bikeshare

# convert Data
data$season <- as.factor(data$season)
data$hr <- as.numeric(data$hr)
```

```r
data$holiday <- as.factor(data$holiday)
data$weekday <- as.factor(data$weekday)
data$workingday <- as.factor(data$workingday)

# Standardize the Data
numeric_columns <- sapply(data, is.numeric)
for (col in names(df)[numeric_columns]) {
  mean_val <- mean(df[[col]])
  sd_val <- sd(df[[col]])
  df[[col]] <- (df[[col]] - mean_val) / sd_val
}

#Train Linear Model
set.seed(1)
train <- sample(nrow(data), 0.8 * nrow(data))
train_set <- data[train, ]
test_set <- data[-train, ]
bikes.lm <- lm(bikers ~ . - casual - registered, data = train_set)
summary(bikes.lm)
```

```
##
## Call:
## lm(formula = bikers ~ . - casual - registered, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -246.80  -70.67  -20.20   44.28  430.19
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 10.0410     9.1081   1.102 0.270318
## season2                     16.9472     7.8435   2.161 0.030755 *
## season3                     29.1898     9.1912   3.176 0.001501 **
## season4                     48.8306     7.7974   6.262 4.02e-10 ***
## mnthFeb                     -2.8144     7.6531  -0.368 0.713073
## mnthMarch                   -1.6140    10.8125  -0.149 0.881345
## mnthApril                   11.1354    16.4281   0.678 0.497903
## mnthMay                     43.1260    20.1609   2.139 0.032463 *
## mnthJune                    11.4570    23.9704   0.478 0.632691
## mnthJuly                   -18.4009    28.4856  -0.646 0.518318
## mnthAug                      7.2090    32.2882   0.223 0.823331
## mnthSept                    54.4690    35.9716   1.514 0.130016
## mnthOct                     70.4675    40.2666   1.750 0.080158 .
## mnthNov                     66.8146    44.4811   1.502 0.133120
## mnthDec                     81.2873    48.3852   1.680 0.093002 .
## day                         -0.2372     0.1430  -1.659 0.097109 .
## hr                           5.6460     0.1925  29.336  < 2e-16 ***
## holiday1                   -22.4822     7.7981  -2.883 0.003951 **
## weekday1                     1.3630     4.7846   0.285 0.775757
## weekday2                    -0.3855     4.7092  -0.082 0.934762
## weekday3                    -1.2618     4.7551  -0.265 0.790749
## weekday4                    -9.0617     4.7160  -1.922 0.054710 .
## weekday5                     5.2910     4.6940   1.127 0.259705
```

```
## weekday6                       5.3833     4.6597   1.155 0.248015
## workingday1                        NA         NA      NA       NA
## weathersitcloudy/misty         7.3621     3.0810   2.390 0.016897 *
## weathersitlight rain/snow    -21.8856     5.0174  -4.362 1.31e-05 ***
## weathersitheavy rain/snow      9.9748   103.8335   0.096 0.923471
## temp                         209.9153    59.0607   3.554 0.000382 ***
## atemp                        109.6514    62.6714   1.750 0.080228 .
## hum                         -160.5908     8.2685 -19.422  < 2e-16 ***
## windspeed                     27.4018    11.6728   2.347 0.018929 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 103.6 on 6885 degrees of freedom
## Multiple R-squared:  0.4078, Adjusted R-squared:  0.4052
## F-statistic: 158.1 on 30 and 6885 DF,  p-value: < 2.2e-16
```

```
lm.model <- stepAIC(bikes.lm, direction = "both")
```

```
## Start:  AIC=64217.57
## bikers ~ (season + mnth + day + hr + holiday + weekday + workingday +
##     weathersit + temp + atemp + hum + windspeed + casual + registered) -
##     casual - registered
##
##
## Step:  AIC=64217.57
## bikers ~ season + mnth + day + hr + holiday + weekday + weathersit +
##     temp + atemp + hum + windspeed
##
##              Df Sum of Sq      RSS   AIC
## <none>                    73882885 64218
## - day         1     29544 73912429 64218
## - weekday     6    139058 74021944 64219
## - atemp       1     32850 73915735 64219
## - windspeed   1     59135 73942021 64221
## - holiday     1     89194 73972079 64224
## - temp        1    135560 74018445 64228
## - weathersit  3    361153 74244039 64245
## - season      3    422999 74305885 64251
## - mnth       11   1291592 75174478 64315
## - hum         1   4047910 77930795 64584
## - hr          1   9234864 83117749 65030
```

```
summary(lm.model)
```

```
##
## Call:
## lm(formula = bikers ~ season + mnth + day + hr + holiday + weekday +
##     weathersit + temp + atemp + hum + windspeed, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -246.80  -70.67  -20.20   44.28  430.19
##
```

```
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  10.0410     9.1081   1.102 0.270318
## season2                      16.9472     7.8435   2.161 0.030755 *
## season3                      29.1898     9.1912   3.176 0.001501 **
## season4                      48.8306     7.7974   6.262 4.02e-10 ***
## mnthFeb                      -2.8144     7.6531  -0.368 0.713073
## mnthMarch                    -1.6140    10.8125  -0.149 0.881345
## mnthApril                    11.1354    16.4281   0.678 0.497903
## mnthMay                      43.1260    20.1609   2.139 0.032463 *
## mnthJune                     11.4570    23.9704   0.478 0.632691
## mnthJuly                    -18.4009    28.4856  -0.646 0.518318
## mnthAug                       7.2090    32.2882   0.223 0.823331
## mnthSept                     54.4690    35.9716   1.514 0.130016
## mnthOct                      70.4675    40.2666   1.750 0.080158 .
## mnthNov                      66.8146    44.4811   1.502 0.133120
## mnthDec                      81.2873    48.3852   1.680 0.093002 .
## day                          -0.2372     0.1430  -1.659 0.097109 .
## hr                            5.6460     0.1925  29.336  < 2e-16 ***
## holiday1                    -22.4822     7.7981  -2.883 0.003951 **
## weekday1                      1.3630     4.7846   0.285 0.775757
## weekday2                     -0.3855     4.7092  -0.082 0.934762
## weekday3                     -1.2618     4.7551  -0.265 0.790749
## weekday4                     -9.0617     4.7160  -1.922 0.054710 .
## weekday5                      5.2910     4.6940   1.127 0.259705
## weekday6                      5.3833     4.6597   1.155 0.248015
## weathersitcloudy/misty        7.3621     3.0810   2.390 0.016897 *
## weathersitlight rain/snow   -21.8856     5.0174  -4.362 1.31e-05 ***
## weathersitheavy rain/snow     9.9748   103.8335   0.096 0.923471
## temp                        209.9153    59.0607   3.554 0.000382 ***
## atemp                       109.6514    62.6714   1.750 0.080228 .
## hum                        -160.5908     8.2685 -19.422  < 2e-16 ***
## windspeed                    27.4018    11.6728   2.347 0.018929 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 103.6 on 6885 degrees of freedom
## Multiple R-squared:  0.4078, Adjusted R-squared:  0.4052
## F-statistic: 158.1 on 30 and 6885 DF,  p-value: < 2.2e-16
```

```r
#Test Model
predict_test <- predict(lm.model, newdata = test_set)
rmse <- sqrt(mean((test_set$bikers - predict_test)^2))
rmse
```

```
## [1] 101.5678
```

Question 3

Explain your approach in 1, including how you coded variables (as numeric or factor), variable selection procedure, and training/test splits you decided to use.

In question one we first changed the data frame to ensure that our data was being interpreted correctly. This involved converting hours to numeric, and switching "holiday, weekday, workingay, and season" to factor variables with corresponding levels. The we converted variables to factor if their levels were not continuously defined. For example, season was coded with numbers, but these numbers are only defined when they are integers, so we converted the variables to make sure there were no ill-defined outcomes. Our process was to train a linear model using our training data and then perform subsset selection using both forward and backward step wise selection to decide which variables were most important. We ended up using the validation set approach to test our method, withholding 20 percent of out data in order to test our model's predictive power.

Question 4

Using a random forest, predict bikers with the other variables in Bikeshare (excluding casual and registered). Make sure to use the variables in their appropriate form (i.e. numeric or factors as relevant to the context of the problem.) Perform model tuning to get a strong predictive model. Use training/test/validation splits of your choice and report a RMSE that describes your model's performance.

```
# Train Random FOrest
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(1)
train <- sample(nrow(data), 0.8 * nrow(data))
train_set <- data[train, ]
test_set <- data[-train, ]
bikes.rf <- randomForest(bikers ~ . - casual - registered, data = train_set, mtry = 6, importance = TRUI

#Test Random Forest
test_predict <- predict(bikes.rf, newdata = test_set)
rmse <- sqrt(mean((test_set$bikers - test_predict)^2))
rmse
```

```
## [1] 37.18259
```

```r
# Pruning/Cross Validation

### Define the control parameters for cross-validation
ctrl <- trainControl(method = "cv", number = 5, verboseIter = FALSE)  # 5-fold cross-validation

### Train the random forest model using cross-validation
model.rf2 <- train(bikers ~ . - casual - registered, data = train_set, method = "rf", trControl = ctrl)

### Access the cross-validation results
cv_results <- model.rf2$results
cv_results
```

```
##   mtry     RMSE  Rsquared      MAE   RMSESD  RsquaredSD     MAESD
## 1    2 94.64802 0.5878814 71.11060 1.194627 0.023456148 0.4909656
## 2   16 36.43452 0.9288495 23.81025 2.067327 0.010690141 0.7116789
## 3   31 34.47204 0.9341856 21.68906 1.269823 0.007885316 0.5087632
```

```r
#Test Random Forest
test_predict <- predict(model.rf2, newdata = test_set)
rmse2 <- sqrt(mean((test_set$bikers - test_predict)^2))
rmse2
```

```
## [1] 37.5201
```

Question 5

Explain your approach in 3, including how you coded variables (as numeric or factor), variable selection procedure, and training/test splits you decided to use.

For our random forest model, we retained the same data frame transformations that we had put in place in question 1. For this model, we first trained the random forest on the traing data, setting our "mtry" to 6, meaning that we restricted our split decision to subsets of 6 features for each split in each tree. We then trained another random forest using 5-fold cross validation in order to ensure better tune our model in an attempt to improve our models predictive power. In training our random forest, we used essentially the same training and test set as was used in our linear model.

Question 6

Plot variable importances of your final random forest. Compare these with the linear regression output. Does the random forest find the same variables to be important as the linear regression?

```r
importance(bikes.rf)
```

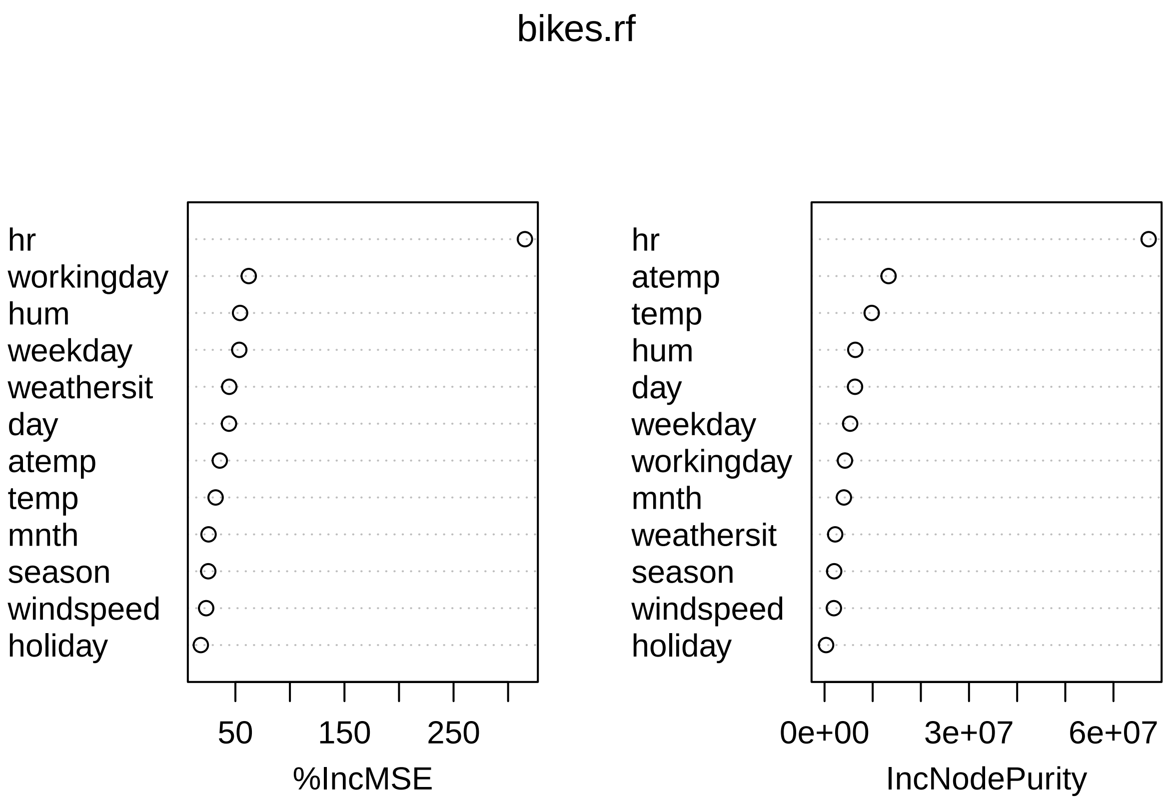
```
##               %IncMSE IncNodePurity
## season      25.11616     2001296.9
## mnth        25.39012     4022040.1
## day         44.14170     6329836.2
## hr         315.40463    67307635.9
## holiday     18.30377      320532.2
## weekday     53.57751     5323534.8
## workingday  62.23800     4237916.8
```

```
## weathersit  44.36976     2206503.3
## temp        31.88275     9792371.1
## atemp       35.71790    13294070.6
## hum         54.30114     6382761.2
## windspeed   23.21635     1938222.9
```

```
bikes.lm
```

```
##
## Call:
## lm(formula = bikers ~ . - casual - registered, data = train_set)
##
## Coefficients:
##              (Intercept)                    season2
##                  10.0410                    16.9472
##                  season3                    season4
##                  29.1898                    48.8306
##                  mnthFeb                  mnthMarch
##                  -2.8144                    -1.6140
##                mnthApril                    mnthMay
##                  11.1354                    43.1260
##                 mnthJune                   mnthJuly
##                  11.4570                   -18.4009
##                  mnthAug                   mnthSept
##                   7.2090                    54.4690
##                  mnthOct                    mnthNov
##                  70.4675                    66.8146
##                  mnthDec                        day
##                  81.2873                    -0.2372
##                       hr                   holiday1
##                   5.6460                   -22.4822
##                 weekday1                   weekday2
##                   1.3630                    -0.3855
##                 weekday3                   weekday4
##                  -1.2618                    -9.0617
##                 weekday5                   weekday6
##                   5.2910                     5.3833
##              workingday1     weathersitcloudy/misty
##                       NA                     7.3621
## weathersitlight rain/snow  weathersitheavy rain/snow
##                 -21.8856                     9.9748
##                     temp                      atemp
##                 209.9153                   109.6514
##                      hum                  windspeed
##                -160.5908                    27.4018
```

```
varImpPlot(bikes.rf)
```



Both linear regression and random variables find the "hr" variable(s) to be important, From there on, it is tough to distinguish which variables are most influential in both models considering becuase the coeficients for the linear model tend to be clustered around similar values, and in similar fashion, the measured importance of the variables in the random forest are nearly idenitical.