

Read Data

11/23/2021

```
library(foreign)
```

```
## Warning: package 'foreign' was built under R version 4.0.3
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## -- Attaching packages ----- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5      v purrr   0.3.4
```

```
## v tibble  3.1.4      v dplyr   1.0.7
```

```
## v tidyr   1.1.3      v stringr 1.4.0
```

```
## v readr   2.0.1      v forcats 0.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.4
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'purrr' was built under R version 4.0.3
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'stringr' was built under R version 4.0.4
```

```
## Warning: package 'forcats' was built under R version 4.0.4
```

```
## -- Conflicts ----- tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()     masks stats::lag()
```

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.0.5
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.0.5
```

```
## Registered S3 method overwritten by 'tree':  
##   method      from  
##   print.tree cli
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.0.5
```

```
## Loaded gbm 2.1.8
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.5
```

```
## randomForest 4.6-14
```

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

```
##  
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':  
##  
##   combine
```

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

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 4.0.5
```

```
##  
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':  
##  
##   lift
```

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(tidyverse)
library(patchwork)
```

```
## Warning: package 'patchwork' was built under R version 4.0.5
```

```
library(UBL)
```

```
## Warning: package 'UBL' was built under R version 4.0.5
```

```
## Loading required package: MBA
```

```
## Warning: package 'MBA' was built under R version 4.0.5
```

```
## Loading required package: gstat
```

```
## Warning: package 'gstat' was built under R version 4.0.5
```

```
## Loading required package: automap
```

```
## Warning: package 'automap' was built under R version 4.0.5
```

```
## Loading required package: sp
```

```
## Warning: package 'sp' was built under R version 4.0.5
```

```
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.0.4
```

```
##
```

```
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
```

```
##
```

```
##      discard
```

```
## The following object is masked from 'package:readr':
```

```
##
```

```
##      col_factor
```

```

sesame <- read.dta("sesame.dta")
sesame <- sesame %>%
  mutate(site=factor(site)) %>%
  mutate(bodyDiff = postbody - prebody,
         letDiff = postlet - prelet,
         formDiff = postform - preform,
         numbDiff = postnumb - prenumb,
         relatDiff = postrelat - prerelat,
         clasfDiff = postclasf - preclasf)
sesame.sd <- sesame%>%
  mutate(sd_pBod = scale(prebody, center = TRUE, scale = TRUE),
         sd_plet = scale(prelet, center = TRUE, scale = TRUE),
         sd_pform = scale(preform, center = TRUE, scale = TRUE),
         sd_pnumb = scale(prenumb, center = TRUE, scale = TRUE),
         sd_prelat = scale(prerelat, center = TRUE, scale = TRUE),
         sd_pclasf = scale(preclasf, center = TRUE, scale = TRUE),
         sd_peabody = scale(peabody, center = TRUE, scale = TRUE),
         sd_age = scale(age, center = TRUE, scale = TRUE),
         male=if_else(sex==1, 1, 0),
         female=if_else(sex==2, 1, 0))

```

Exploratory Data Analysis

```
head(sesame)
```

```

##   rownames id site sex age viewcat setting viewenc prebody prelet preform
## 1      1  1  1  1  66      1      2      1     16     23     12
## 2      2  2  1  2  67      3      2      1     30     26      9
## 3      3  3  1  1  56      3      2      2     22     14      9
## 4      4  4  1  1  49      1      2      2     23     11     10
## 5      5  5  1  1  69      4      2      2     32     47     15
## 6      6  6  1  2  54      3      2      2     29     26     10
##   prenumb prerelat preclasf postbody postlet postform postnumb postrelat
## 1      40      14      20      18      30      14      44      14
## 2      39      16      22      30      37      17      39      14
## 3       9       9       8      21      46      15      40       9
## 4      14       9      13      21      14      13      19       8
## 5      51      17      22      32      63      18      54      14
## 6      33      14      14      27      36      14      39      16
##   postclasf peabody agecat encour _Isite_2 _Isite_3 _Isite_4 _Isite_5 regular
## 1      23      62      1      1      0      0      0      0      0
## 2      22       8      1      1      0      0      0      0      1
## 3      19      32      1      0      0      0      0      0      1
## 4      15      27      0      0      0      0      0      0      0
## 5      21      71      1      0      0      0      0      0      1
## 6      24      32      1      0      0      0      0      0      1
##   bodyDiff letDiff formDiff numbDiff relatDiff clasfDiff
## 1       2       7       2       4       0       3
## 2       0      11       8       0      -2       0
## 3      -1      32       6      31       0      11
## 4      -2       3       3       5      -1       2

```

## 5	0	16	3	3	-3	-1
## 6	-2	10	4	6	2	10

Variables:

The ID refers to a subject's identification number. The site refers to the age and background information of the child. A site value of 1 indicates a 3-5 year old disadvantaged child from the inner city. A site value of 2 represents a 4 year old advantaged child from the suburbs. A value of 3 represents an advantaged rural child. A site value of 4 indicates a disadvantaged rural child. Lastly, a value of 5 represents a disadvantaged Spanish speaking child. For the sex, a value of 1 indicates male, and a value of 2 indicates female. The age category is the child's age in months. The viewcat column is the frequency of viewing Sesame Street (1 = rarely, 2 = once/twice per week, 3 = 3-5 times a week, 4 = more than 5 times per week). The setting is where Sesame Street was viewed; a value of 1 indicates home and a value of 2 indicates school. The viewenc column refers to if the child was encouraged to watch or not (1 = child not encouraged, 2 = child encouraged). Encour is the same variable but with values 0 and 1, respectively. Regular is an indicator variable representing if a child is a regular viewer (0 = rarely watched, 1 = watched once per week or greater).

The prebody, prelet, preform, prenumb, prerelat, and preclasf columns all describe pretest scores on varying types of assessments (body parts, letters, forms, numbers, relational terms, and classification skills, respectively). The columns labelled postbody, postlet, postform, postnumb, postrelat, and postclasf are the children's respective posttest scores. Above, we created the following variables - bodyDiff, letDiff, formDiff, numbDiff, relatDiff, clasfDiff - to represent the difference in posttest scores and pretest scores for each child. Lastly, peabody represents a score of "mental age" for vocabulary maturity from the Peabody Picture Vocabulary Test.

Our main focus will be on the new variables we created (bodyDiff, letDiff, formDiff, numbDiff, relatDiff, clasfDiff) and variables related to how often the children watch Sesame Street (namely, viewcat and regular). Lastly, we will look into the backgrounds of the children, including site, sex, and age.

Distributions:

For the purposes of our analysis, we will first look at the distributions of bodyDiff, letDiff, formDiff, numbDiff, relatDiff, and clasfDiff.

#want to visualize distributions of bodyDiff, letDiff, formDiff, numbDiff, relatDiff, clasfDiff

```
bodyDiffplot <- ggplot(sesame, aes(x = bodyDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of bodyDiff", x = "Post - Pre on Body Parts", y = "Count") +
  theme_minimal()

letDiffplot <- ggplot(sesame, aes(x = letDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of letDiff", x = "Post - Pre on Letters", y = "Count") +
  theme_minimal()

formDiffplot <- ggplot(sesame, aes(x = formDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of formDiff", x = "Post - Pre on Forms", y = "Count") +
  theme_minimal()

numbDiffplot <- ggplot(sesame, aes(x = numbDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of numbDiff", x = "Post - Pre on Numbers", y = "Count") +
```

```

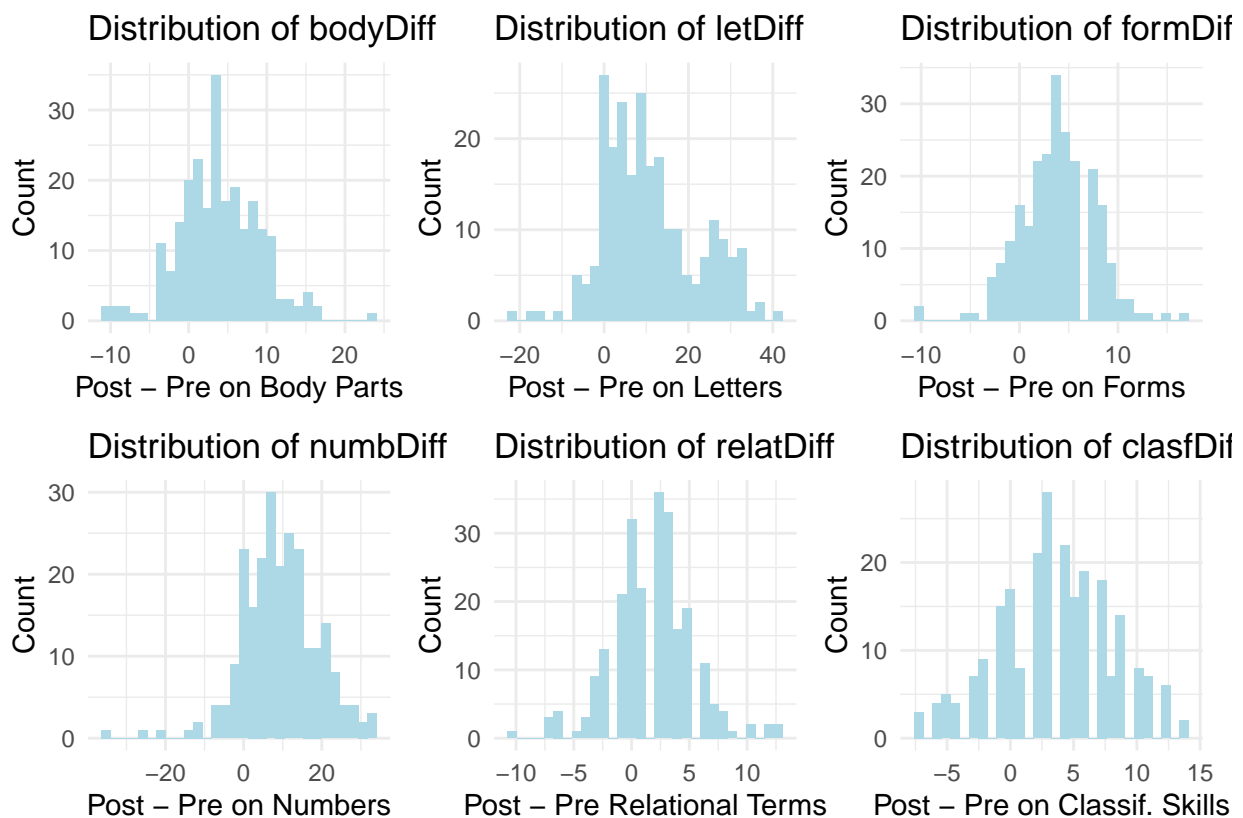
theme_minimal()

relatDiffplot <- ggplot(sesame, aes(x = relatDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of relatDiff", x = "Post - Pre Relational Terms", y = "Count") +
  theme_minimal()

clasfDiffplot <- ggplot(sesame, aes(x = clasfDiff)) +
  geom_histogram(fill = "lightblue") +
  labs(title = "Distribution of clasfDiff", x = "Post - Pre on Classif. Skills", y = "Count") +
  theme_minimal()

bodyDiffplot + letDiffplot + formDiffplot + numbDiffplot + relatDiffplot + clasfDiffplot

```



The six variables above were calculated by subtracting pre-test scores from post-test scores, so they are all numerical. The distributions of these six variables (bodyDiff, letDiff, formDiff, numbDiff, relatDiff, and clasfDiff) all appear to be roughly normal and unimodal. BodyDiff, letDiff, formDiff, relatDiff, and classDiff do not appear to have any obvious extreme outliers. Numbdiff, however, seems to be slightly left-skewed with outliers to the left -20. All of the six variables appear to have centers between 2 and 4.

We will now examine the distributions of the variables related to how often children watch Sesame Street (namely, viewcat and regular).

```

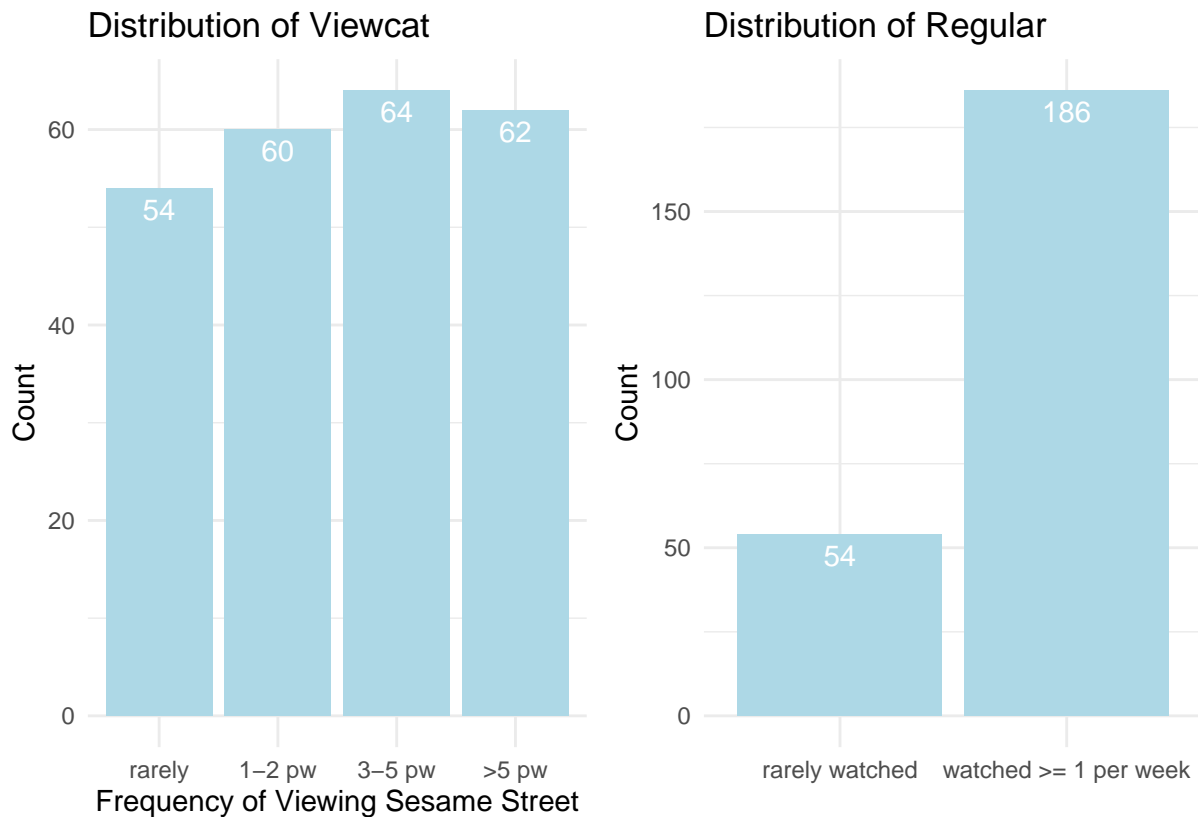
# want to visualize distributions of viewcat and regular

```

```
viewcatplot <- ggplot(sesame, aes(x = factor(viewcat))) +
  geom_bar(fill = "lightblue") +
  labs(title = "Distribution of Viewcat", x = "Frequency of viewing Sesame Street", y = "Count") +
  scale_x_discrete("Frequency of Viewing Sesame Street", labels=c("rarely", "1-2 pw", "3-5 pw", ">5 pw")) +
  theme_minimal() +
  geom_text(aes(label = ..count..), stat = "count", vjust = 1.5, colour = "white")

regularplot <- ggplot(sesame, aes(x = factor(regular))) +
  geom_bar(fill = "lightblue") +
  labs(title = "Distribution of Regular", y = "Count") +
  scale_x_discrete(labels=c("rarely watched", "watched >= 1 per week")) +
  theme_minimal() +
  theme(axis.title.x = element_blank()) +
  geom_text(aes(label = ..count..), stat = "count", vjust = 1.5, colour = "white")

viewcatplot + regularplot
```



Both of these variables are categorical. On the left, viewcat appears to have a roughly uniform distribution, with “rarely” having the least amount of children and 3-5 times per week having the most (the range is only 10 children, so all of the bars are relatively close in height). For the variable regular, the category “watched once per week or greater” has far more observations than “rarely watched.” The former category has more than triple the amount of the latter. We will be aware of this disparity in our analysis and continue with caution towards potential bias.

Lastly, we want to examine the distributions of site, sex, and age, all variables that relate to a child’s

background.

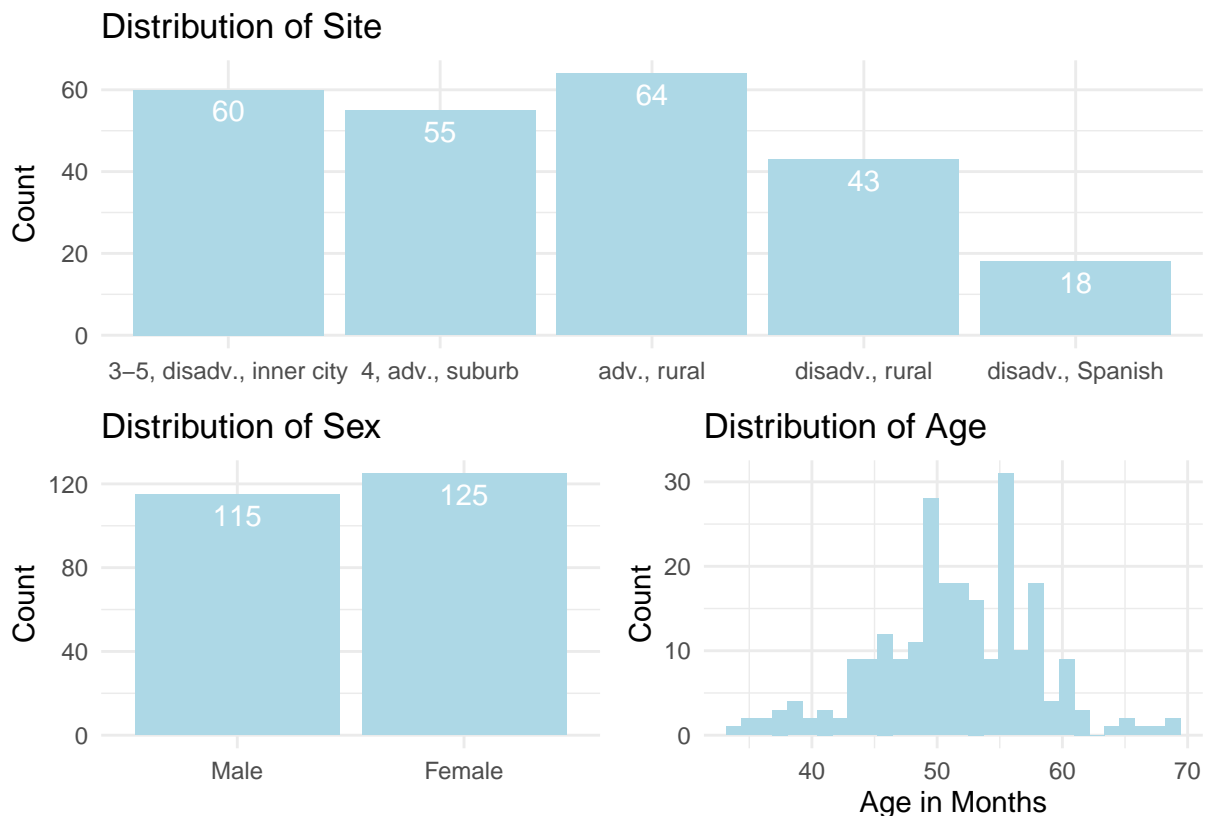
```
# want to visualize distributions of site, sex, and age
```

```
siteplot <- ggplot(sesame, aes(x = factor(site))) +  
  geom_bar(fill = "lightblue") +  
  labs(title = "Distribution of Site", y = "Count") +  
  scale_x_discrete(labels=c("3-5, disadv., inner city", "4, adv., suburb", "adv., rural", "disadv., rural", "disadv., Spanish")) +  
  theme_minimal() +  
  theme(axis.title.x = element_blank()) +  
  geom_text(aes(label = ..count..), stat = "count", vjust = 1.5, colour = "white")
```

```
sexplot <- ggplot(sesame, aes(x = factor(sex))) +  
  geom_bar(fill = "lightblue") +  
  labs(title = "Distribution of Sex", y = "Count") +  
  scale_x_discrete(labels=c("Male", "Female")) +  
  theme_minimal() +  
  theme(axis.title.x = element_blank()) +  
  geom_text(aes(label = ..count..), stat = "count", vjust = 1.5, colour = "white")
```

```
ageplot <- ggplot(sesame, aes(x = age)) +  
  geom_histogram(fill = "lightblue") +  
  labs(title = "Distribution of Age", x = "Age in Months", y = "Count") +  
  theme_minimal()
```

```
siteplot / (sexplot + ageplot)
```



Site and Sex are both categorical variables. Distribution of Site has four categories with roughly the same amount of children (ranging from 43 to 64), but one category with far fewer observations (disadvantaged Spanish-speaking). This category has less than half of the observations as the next smallest category, which is a relatively large disparity. We will continue our analysis with caution towards this bias in the data. The distribution of sex is very even - the male category has 115 observations, while the female category has 125 observations. Age is a numerical variable that appears to be normal and bimodal, with two peaks around 50 and 56. There do not appear to be any extreme outliers in the distribution of age.

Q.1 Prediction Question: Can we use linear regression to predict the change in a child's test scores that occur after watching Sesame street (or in some instances, not watching Sesame street)?

Linear Regression Models

Here, I am fitting 6 linear regression models. Each of the models predicts a different difference in test score.

```
sesame.q1 <- sesame

sesame.q1$site <- as.factor(sesame.q1$site)
sesame.q1$sex <- as.factor(sesame.q1$sex)
sesame.q1$viewcat <- as.factor(sesame.q1$viewcat)
sesame.q1$setting <- as.factor(sesame.q1$setting)
sesame.q1$viewenc <- as.factor(sesame.q1$viewenc)

# Scaling Variables

sesame.q1$bodyDiff <- rescale(sesame.q1$bodyDiff, to = c(0, 30))
sesame.q1$letDiff <- rescale(sesame.q1$letDiff, to = c(0, 30))
sesame.q1$formDiff <- rescale(sesame.q1$formDiff, to = c(0, 30))
sesame.q1$numbDiff <- rescale(sesame.q1$numbDiff, to = c(0, 30))
sesame.q1$relatDiff <- rescale(sesame.q1$relatDiff, to = c(0, 30))
sesame.q1$clasfDiff <- rescale(sesame.q1$clasfDiff, to = c(0, 30))
```

```
# Test-Train Split

set.seed(1)
train <- sample(1:nrow(sesame.q1), nrow(sesame.q1)*0.7)

training = sesame.q1[train,]
testing = sesame.q1[-train, ]
```

Firstly, we factored the following variables to encode them as categoricals: **site**, **sex**, **viewcat**, **setting**, **viewenc**.

One problem that we envisioned when evaluating the and comparing the different models is that the tests are scored on different scales. For example, the scores for the test on knowledge of body parts (noted by **bodyDiff**) range from 0-32, while those of the test on letters (noted by **letDiff**) range from 0-58. To be able to aptly compare the mean squared error (MSE) between models, we also decided to convert each response variable to the same range. More specifically, we scaled each variable to the arbitrary range [0, 30].

Lastly, we randomly split the data between testing and training, using 70% of the data for training and 30% of the data for testing.

```
lin.mod1.full <- lm(bodyDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)
summary(lin.mod1.full)
```

```
##
## Call:
## lm(formula = bodyDiff ~ (site + sex + age + viewcat + setting +
##      viewenc)^2, data = training)
##
## Residuals:
```

	Min	1Q	Median	3Q	Max
	-8.0419	-2.3623	-0.0666	2.0079	9.9599

```
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -5.21460    14.63930  -0.356   0.7224
## site2         -1.67860    13.72456  -0.122   0.9029
## site3          1.18809     9.13599   0.130   0.8968
## site4         36.76705    16.49603   2.229   0.0278 *
## site5          2.51268    17.41166   0.144   0.8855
## sex2           2.67852     6.79950   0.394   0.6944
## age            0.30634     0.26777   1.144   0.2551
## viewcat2       11.62045    14.78702   0.786   0.4336
## viewcat3       12.46797    13.36691   0.933   0.3530
## viewcat4       29.95220    14.54421   2.059   0.0418 *
## setting2       -9.74959     9.11761  -1.069   0.2872
## viewenc2       19.16298    10.20959   1.877   0.0631 .
## site2:sex2      3.01414     2.22656   1.354   0.1786
## site3:sex2      2.32383     2.17020   1.071   0.2866
## site4:sex2      0.53119     2.63525   0.202   0.8406
## site5:sex2      0.32139     3.71691   0.086   0.9312
## site2:age       0.04657     0.24040   0.194   0.8467
## site3:age       0.03300     0.17150   0.192   0.8477
## site4:age      -0.59988     0.30613  -1.960   0.0525 .
## site5:age       0.01087     0.33703   0.032   0.9743
## site2:viewcat2   3.94606     5.50815   0.716   0.4752
## site3:viewcat2   2.11625     3.88814   0.544   0.5873
## site4:viewcat2   2.33983     3.90246   0.600   0.5500
## site5:viewcat2   5.15446     6.28877   0.820   0.4142
## site2:viewcat3  -4.60122     5.15701  -0.892   0.3742
## site3:viewcat3  -2.61690     3.66597  -0.714   0.4768
## site4:viewcat3  -4.29765     4.15537  -1.034   0.3033
## site5:viewcat3   NA          NA        NA        NA
## site2:viewcat4  -2.76550     5.23768  -0.528   0.5985
## site3:viewcat4  -2.14835     3.84372  -0.559   0.5773
## site4:viewcat4  -6.19787     4.85958  -1.275   0.2048
## site5:viewcat4  -3.85743     6.63742  -0.581   0.5623
## site2:setting2   0.63008     2.76440   0.228   0.8201
## site3:setting2   0.14259     3.16610   0.045   0.9642
## site4:setting2   5.09348     3.25326   1.566   0.1203
## site5:setting2   NA          NA        NA        NA
## site2:viewenc2   0.89088     2.60475   0.342   0.7330
## site3:viewenc2  -1.79090     3.30478  -0.542   0.5890
## site4:viewenc2  -5.97857     3.03298  -1.971   0.0512 .
```

```
## site5:viewenc2    -2.07853    7.09228   -0.293    0.7700
## sex2:age          -0.10641    0.12653   -0.841    0.4022
## sex2:viewcat2     1.49879    2.64935    0.566    0.5727
## sex2:viewcat3     0.58754    2.51057    0.234    0.8154
## sex2:viewcat4    -1.20166    2.68241   -0.448    0.6550
## sex2:setting2    -1.53778    1.70350   -0.903    0.3686
## sex2:viewenc2     1.20679    1.93998    0.622    0.5352
## age:viewcat2     -0.27426    0.26179   -1.048    0.2971
## age:viewcat3     -0.15379    0.23654   -0.650    0.5169
## age:viewcat4     -0.46253    0.26054   -1.775    0.0786 .
## age:setting2      0.05401    0.15397    0.351    0.7264
## age:viewenc2     -0.27988    0.18128   -1.544    0.1254
## viewcat2:setting2  5.64914    3.31141    1.706    0.0908 .
## viewcat3:setting2  6.07677    3.20554    1.896    0.0606 .
## viewcat4:setting2  6.21301    3.52866    1.761    0.0810 .
## viewcat2:viewenc2 -7.71647    3.54437   -2.177    0.0316 *
## viewcat3:viewenc2 -6.24879    3.34488   -1.868    0.0644 .
## viewcat4:viewenc2 -7.65623    3.62444   -2.112    0.0369 *
## setting2:viewenc2  1.97122    2.32061    0.849    0.3974
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.226 on 112 degrees of freedom
## Multiple R-squared:  0.4033, Adjusted R-squared:  0.1103
## F-statistic: 1.377 on 55 and 112 DF,  p-value: 0.07819
```

```
AIC(lin.mod1.full)
```

```
## [1] 1006.935
```

```
yhat <- predict(lin.mod1.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod1.full, newdata = testing): prediction from a rank-
## deficient fit may be misleading
```

```
y.test <- testing[, "bodyDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 32.44637
```

```
lin.mod2.full <- lm(letDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)
summary(lin.mod2.full)
```

```
##
## Call:
## lm(formula = letDiff ~ (site + sex + age + viewcat + setting +
##      viewenc)^2, data = training)
##
## Residuals:
```

```

##      Min      1Q   Median      3Q      Max
## -12.7462 -2.3134  0.0386  2.3986 10.2234
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    19.20505   15.54400    1.236  0.2192
## site2          -7.55435   14.57273   -0.518  0.6052
## site3           7.76205    9.70058    0.800  0.4253
## site4        -12.82246   17.51547   -0.732  0.4657
## site5           1.39944   18.48769    0.076  0.9398
## sex2           -7.03111    7.21970   -0.974  0.3322
## age            -0.13633    0.28432   -0.480  0.6325
## viewcat2       -20.28956   15.70085   -1.292  0.1989
## viewcat3        -0.74326   14.19298   -0.052  0.9583
## viewcat4         5.71015   15.44303    0.370  0.7123
## setting2        -0.93795    9.68107   -0.097  0.9230
## viewenc2        -5.41528   10.84054   -0.500  0.6184
## site2:sex2       1.20390    2.36416    0.509  0.6116
## site3:sex2      -0.63445    2.30431   -0.275  0.7836
## site4:sex2      -3.21100    2.79810   -1.148  0.2536
## site5:sex2      -1.80949    3.94661   -0.458  0.6475
## site2:age        0.27245    0.25526    1.067  0.2881
## site3:age       -0.17713    0.18209   -0.973  0.3328
## site4:age        0.32387    0.32505    0.996  0.3212
## site5:age       -0.06901    0.35786   -0.193  0.8474
## site2:viewcat2  -1.69100    5.84855   -0.289  0.7730
## site3:viewcat2   0.28314    4.12842    0.069  0.9454
## site4:viewcat2   3.57460    4.14363    0.863  0.3902
## site5:viewcat2   4.79038    6.67741    0.717  0.4746
## site2:viewcat3  -5.13072    5.47571   -0.937  0.3508
## site3:viewcat3  -1.48691    3.89252   -0.382  0.7032
## site4:viewcat3  -5.03691    4.41217   -1.142  0.2561
## site5:viewcat3      NA         NA         NA      NA
## site2:viewcat4  -3.19663    5.56136   -0.575  0.5666
## site3:viewcat4  -0.93894    4.08126   -0.230  0.8185
## site4:viewcat4  -5.99748    5.15990   -1.162  0.2476
## site5:viewcat4  -0.21046    7.04760   -0.030  0.9762
## site2:setting2  -1.36444    2.93524   -0.465  0.6429
## site3:setting2  -1.30832    3.36176   -0.389  0.6979
## site4:setting2  -4.68028    3.45430   -1.355  0.1782
## site5:setting2      NA         NA         NA      NA
## site2:viewenc2  -2.65415    2.76572   -0.960  0.3393
## site3:viewenc2   0.03940    3.50901    0.011  0.9911
## site4:viewenc2  -0.05966    3.22042   -0.019  0.9853
## site5:viewenc2   7.97522    7.53058    1.059  0.2919
## sex2:age        0.15410    0.13435    1.147  0.2538
## sex2:viewcat2    2.86176    2.81308    1.017  0.3112
## sex2:viewcat3   -0.20025    2.66573   -0.075  0.9403
## sex2:viewcat4   -0.45238    2.84818   -0.159  0.8741
## sex2:setting2   -1.93840    1.80878   -1.072  0.2862
## sex2:viewenc2    3.18303    2.05987    1.545  0.1251
## age:viewcat2     0.38696    0.27796    1.392  0.1666
## age:viewcat3     0.14608    0.25116    0.582  0.5620
## age:viewcat4     0.01779    0.27664    0.064  0.9488

```

```
## age:setting2      0.01359    0.16348    0.083    0.9339
## age:viewenc2      0.02087    0.19248    0.108    0.9139
## viewcat2:setting2  2.59739    3.51606    0.739    0.4616
## viewcat3:setting2  4.02713    3.40364    1.183    0.2392
## viewcat4:setting2 -0.78643    3.74673   -0.210    0.8341
## viewcat2:viewenc2 -3.47489    3.76341   -0.923    0.3578
## viewcat3:viewenc2 -1.80040    3.55159   -0.507    0.6132
## viewcat4:viewenc2  1.81492    3.84842    0.472    0.6381
## setting2:viewenc2  4.70866    2.46403    1.911    0.0586 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.488 on 112 degrees of freedom
## Multiple R-squared:  0.5727, Adjusted R-squared:  0.3628
## F-statistic: 2.729 on 55 and 112 DF,  p-value: 3.56e-06
```

```
AIC(lin.mod2.full)
```

```
## [1] 1027.083
```

```
yhat <- predict(lin.mod2.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod2.full, newdata = testing): prediction from a rank-
## deficient fit may be misleading
```

```
y.test <- testing[, "letDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 24.44647
```

```
lin.mod3.full <- lm(formDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)
summary(lin.mod3.full)
```

```
##
## Call:
## lm(formula = formDiff ~ (site + sex + age + viewcat + setting +
##      viewenc)^2, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -8.9228 -1.8652 -0.0169  1.8737 12.7649
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   29.944763  14.480042   2.068   0.0409 *
## site2         12.327359   13.575252   0.908   0.3658
## site3        -1.058753    9.036594  -0.117   0.9069
## site4         4.392580   16.316566   0.269   0.7883
## site5         4.356217   17.222238   0.253   0.8008
```

## sex2	3.675853	6.725528	0.547	0.5858
## age	-0.296259	0.264860	-1.119	0.2657
## viewcat2	-24.508158	14.626151	-1.676	0.0966 .
## viewcat3	-15.832596	13.221492	-1.197	0.2336
## viewcat4	-2.141958	14.385978	-0.149	0.8819
## setting2	7.516078	9.018414	0.833	0.4064
## viewenc2	-15.197856	10.098523	-1.505	0.1351
## site2:sex2	-3.772624	2.202334	-1.713	0.0895 .
## site3:sex2	-4.100876	2.146588	-1.910	0.0586 .
## site4:sex2	-5.714483	2.606577	-2.192	0.0304 *
## site5:sex2	-4.513182	3.676472	-1.228	0.2222
## site2:age	-0.271127	0.237788	-1.140	0.2566
## site3:age	0.001186	0.169630	0.007	0.9944
## site4:age	-0.031048	0.302804	-0.103	0.9185
## site5:age	-0.082011	0.333365	-0.246	0.8061
## site2:viewcat2	6.777720	5.448230	1.244	0.2161
## site3:viewcat2	6.728359	3.845840	1.750	0.0829 .
## site4:viewcat2	5.866619	3.860003	1.520	0.1314
## site5:viewcat2	7.030391	6.220350	1.130	0.2608
## site2:viewcat3	2.122722	5.100911	0.416	0.6781
## site3:viewcat3	4.363483	3.626088	1.203	0.2314
## site4:viewcat3	0.418813	4.110165	0.102	0.9190
## site5:viewcat3	NA	NA	NA	NA
## site2:viewcat4	1.747657	5.180694	0.337	0.7365
## site3:viewcat4	5.788361	3.801908	1.522	0.1307
## site4:viewcat4	-0.137820	4.806713	-0.029	0.9772
## site5:viewcat4	1.851415	6.565206	0.282	0.7785
## site2:setting2	1.586718	2.734325	0.580	0.5629
## site3:setting2	-6.267179	3.131653	-2.001	0.0478 *
## site4:setting2	1.393469	3.217863	0.433	0.6658
## site5:setting2	NA	NA	NA	NA
## site2:viewenc2	-0.253451	2.576413	-0.098	0.9218
## site3:viewenc2	4.996265	3.268825	1.528	0.1292
## site4:viewenc2	-4.076868	2.999986	-1.359	0.1769
## site5:viewenc2	0.542663	7.015121	0.077	0.9385
## sex2:age	0.040381	0.125154	0.323	0.7476
## sex2:viewcat2	-2.039436	2.620528	-0.778	0.4381
## sex2:viewcat3	0.317803	2.483262	0.128	0.8984
## sex2:viewcat4	-0.544298	2.653230	-0.205	0.8378
## sex2:setting2	-2.642309	1.684970	-1.568	0.1197
## sex2:viewenc2	-2.157924	1.918879	-1.125	0.2632
## age:viewcat2	0.364615	0.258937	1.408	0.1619
## age:viewcat3	0.253258	0.233965	1.082	0.2814
## age:viewcat4	0.017013	0.257708	0.066	0.9475
## age:setting2	-0.094395	0.152294	-0.620	0.5366
## age:viewenc2	0.288927	0.179305	1.611	0.1099
## viewcat2:setting2	0.045540	3.275387	0.014	0.9889
## viewcat3:setting2	-0.916800	3.170671	-0.289	0.7730
## viewcat4:setting2	2.713885	3.490273	0.778	0.4385
## viewcat2:viewenc2	7.112277	3.505815	2.029	0.0449 *
## viewcat3:viewenc2	2.220835	3.308488	0.671	0.5034
## viewcat4:viewenc2	0.718803	3.585006	0.201	0.8415
## setting2:viewenc2	-1.590335	2.295369	-0.693	0.4898
## ---				

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.18 on 112 degrees of freedom
## Multiple R-squared:  0.3905, Adjusted R-squared:  0.09115
## F-statistic: 1.305 on 55 and 112 DF,  p-value: 0.1189
```

```
AIC(lin.mod3.full)
```

```
## [1] 1003.259
```

```
yhat <- predict(lin.mod3.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod3.full, newdata = testing): prediction from a rank-
## deficient fit may be misleading
```

```
y.test <- testing[, "formDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 23.18844
```

```
lin.mod4.full <- lm(numbDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)
summary(lin.mod4.full)
```

```
##
## Call:
## lm(formula = numbDiff ~ (site + sex + age + viewcat + setting +
##   viewenc)^2, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.9731  -2.2530  -0.0413   2.4864   9.4788
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    16.70541    15.13888   1.103  0.2722
## site2          15.96399    14.19292   1.125  0.2631
## site3           6.64629     9.44776   0.703  0.4832
## site4          -4.69062    17.05897  -0.275  0.7838
## site5           1.27560    18.00585   0.071  0.9436
## sex2            9.80788     7.03154   1.395  0.1658
## age            -0.08799     0.27691  -0.318  0.7513
## viewcat2       -12.34222    15.29164  -0.807  0.4213
## viewcat3       -11.17277    13.82307  -0.808  0.4206
## viewcat4         3.28623    15.04054   0.218  0.8274
## setting2        17.35032     9.42875   1.840  0.0684
## viewenc2        -7.37000    10.55800  -0.698  0.4866
## site2:sex2      -3.34924     2.30254  -1.455  0.1486
## site3:sex2      -2.90207     2.24426  -1.293  0.1986
## site4:sex2      -3.42395     2.72518  -1.256  0.2116
```

```

## site5:sex2      -0.25383    3.84375   -0.066    0.9475
## site2:age       -0.06976    0.24861   -0.281    0.7795
## site3:age       -0.05208    0.17735   -0.294    0.7696
## site4:age        0.27163    0.31658    0.858    0.3927
## site5:age        0.05536    0.34853    0.159    0.8741
## site2:viewcat2  -6.50669    5.69612   -1.142    0.2558
## site3:viewcat2  -1.86918    4.02083   -0.465    0.6429
## site4:viewcat2  -4.52539    4.03563   -1.121    0.2645
## site5:viewcat2  -0.06176    6.50337   -0.009    0.9924
## site2:viewcat3  -6.54853    5.33300   -1.228    0.2220
## site3:viewcat3    0.37710    3.79107    0.099    0.9209
## site4:viewcat3  -2.80346    4.29718   -0.652    0.5155
## site5:viewcat3      NA         NA         NA         NA
## site2:viewcat4 -10.05730    5.41641   -1.857    0.0660 .
## site3:viewcat4    0.44685    3.97489    0.112    0.9107
## site4:viewcat4  -6.50503    5.02542   -1.294    0.1982
## site5:viewcat4  -5.38827    6.86392   -0.785    0.4341
## site2:setting2  -2.14597    2.85874   -0.751    0.4544
## site3:setting2  -4.76023    3.27414   -1.454    0.1488
## site4:setting2  -7.31658    3.36428   -2.175    0.0317 *
## site5:setting2      NA         NA         NA         NA
## site2:viewenc2  -2.72744    2.69364   -1.013    0.3135
## site3:viewenc2  -1.35549    3.41756   -0.397    0.6924
## site4:viewenc2  -1.50418    3.13649   -0.480    0.6325
## site5:viewenc2    6.83681    7.33431    0.932    0.3533
## sex2:age        -0.16226    0.13085   -1.240    0.2176
## sex2:viewcat2     2.21351    2.73976    0.808    0.4208
## sex2:viewcat3     2.04053    2.59625    0.786    0.4336
## sex2:viewcat4    -0.65653    2.77395   -0.237    0.8133
## sex2:setting2    -0.04353    1.76164   -0.025    0.9803
## sex2:viewenc2     1.03717    2.00619    0.517    0.6062
## age:viewcat2     0.29579    0.27072    1.093    0.2769
## age:viewcat3     0.28309    0.24461    1.157    0.2496
## age:viewcat4     0.08420    0.26943    0.312    0.7552
## age:setting2    -0.22758    0.15922   -1.429    0.1557
## age:viewenc2     0.12555    0.18746    0.670    0.5044
## viewcat2:setting2 -2.23219    3.42442   -0.652    0.5158
## viewcat3:setting2 -0.89840    3.31494   -0.271    0.7869
## viewcat4:setting2 -1.33874    3.64908   -0.367    0.7144
## viewcat2:viewenc2  3.29199    3.66533    0.898    0.3710
## viewcat3:viewenc2  1.09871    3.45902    0.318    0.7514
## viewcat4:viewenc2  2.49482    3.74812    0.666    0.5070
## setting2:viewenc2  0.59013    2.39981    0.246    0.8062
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.371 on 112 degrees of freedom
## Multiple R-squared:  0.3558, Adjusted R-squared:  0.03945
## F-statistic: 1.125 on 55 and 112 DF,  p-value: 0.2971

```

```
AIC(lin.mod4.full)
```

```
## [1] 1018.209
```



```
yhat <- predict(lin.mod4.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod4.full, newdata = testing): prediction from a rank-  
## deficient fit may be misleading
```

```
y.test <- testing[, "numbDiff"]
```

```
# Test MSE  
mean((yhat-y.test)^2)
```

```
## [1] 28.23554
```

```
lin.mod5.full <- lm(relatDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)  
summary(lin.mod5.full)
```

```
##  
## Call:  
## lm(formula = relatDiff ~ (site + sex + age + viewcat + setting +  
##      viewenc)^2, data = training)  
##  
## Residuals:  
##      Min       1Q   Median       3Q      Max   
## -10.1381  -2.1057  -0.0037   1.9658  11.4625   
##  
## Coefficients: (2 not defined because of singularities)  
##              Estimate Std. Error t value Pr(>|t|)      
## (Intercept)    17.06032    14.42797   1.182  0.2395      
## site2          -4.32400    13.52644  -0.320  0.7498      
## site3           0.09709     9.00410   0.011  0.9914      
## site4           8.82232    16.25789   0.543  0.5885      
## site5          10.23066    17.16031   0.596  0.5523      
## sex2           11.05162     6.70134   1.649  0.1019      
## age            -0.19817     0.26391  -0.751  0.4543      
## viewcat2       -12.06564    14.57356  -0.828  0.4095      
## viewcat3        -1.76765    13.17395  -0.134  0.8935      
## viewcat4        12.95777    14.33425   0.904  0.3679      
## setting2         5.49314     8.98598   0.611  0.5422      
## viewenc2         1.41758    10.06221   0.141  0.8882      
## site2:sex2      -2.89712     2.19441  -1.320  0.1895      
## site3:sex2      -1.50609     2.13887  -0.704  0.4828      
## site4:sex2      -5.48305     2.59720  -2.111  0.0370 *   
## site5:sex2      -2.72621     3.66325  -0.744  0.4583      
## site2:age        0.14657     0.23693   0.619  0.5374      
## site3:age        0.09339     0.16902   0.553  0.5817      
## site4:age        0.11553     0.30171   0.383  0.7025      
## site5:age       -0.12613     0.33217  -0.380  0.7049      
## site2:viewcat2   1.39325     5.42864   0.257  0.7979      
## site3:viewcat2   0.58876     3.83201   0.154  0.8782      
## site4:viewcat2  -3.40928     3.84612  -0.886  0.3773      
## site5:viewcat2  -0.56097     6.19798  -0.091  0.9280      
## site2:viewcat3  -0.72033     5.08257  -0.142  0.8876
```

```
## site3:viewcat3      -0.90613      3.61305     -0.251      0.8024
## site4:viewcat3      -7.73848      4.09539     -1.890      0.0614 .
## site5:viewcat3              NA              NA              NA              NA
## site2:viewcat4      -3.95053      5.16206     -0.765      0.4457
## site3:viewcat4      -1.08877      3.78824     -0.287      0.7743
## site4:viewcat4      -3.82113      4.78943     -0.798      0.4267
## site5:viewcat4      -1.25414      6.54160     -0.192      0.8483
## site2:setting2      -0.87824      2.72449     -0.322      0.7478
## site3:setting2      -3.25960      3.12039     -1.045      0.2985
## site4:setting2      -6.71576      3.20629     -2.095      0.0385 *
## site5:setting2              NA              NA              NA              NA
## site2:viewenc2       0.09165      2.56715      0.036      0.9716
## site3:viewenc2       0.08926      3.25707      0.027      0.9782
## site4:viewenc2      -5.67745      2.98920     -1.899      0.0601 .
## site5:viewenc2      -1.58031      6.98990     -0.226      0.8215
## sex2:age            -0.10161      0.12470     -0.815      0.4169
## sex2:viewcat2       -0.32485      2.61110     -0.124      0.9012
## sex2:viewcat3       -0.81092      2.47433     -0.328      0.7437
## sex2:viewcat4       -2.75732      2.64369     -1.043      0.2992
## sex2:setting2       -3.19781      1.67891     -1.905      0.0594 .
## sex2:viewenc2       -0.94590      1.91198     -0.495      0.6218
## age:viewcat2         0.29068      0.25801      1.127      0.2623
## age:viewcat3         0.09494      0.23312      0.407      0.6846
## age:viewcat4        -0.11889      0.25678     -0.463      0.6443
## age:setting2        -0.01405      0.15175     -0.093      0.9264
## age:viewenc2         0.06598      0.17866      0.369      0.7126
## viewcat2:setting2   -2.52471      3.26361     -0.774      0.4408
## viewcat3:setting2   -0.09666      3.15927     -0.031      0.9756
## viewcat4:setting2    2.70973      3.47772      0.779      0.4375
## viewcat2:viewenc2   -2.40402      3.49321     -0.688      0.4928
## viewcat3:viewenc2   -1.85347      3.29659     -0.562      0.5751
## viewcat4:viewenc2   -2.73805      3.57212     -0.767      0.4450
## setting2:viewenc2   -3.01217      2.28711     -1.317      0.1905
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.165 on 112 degrees of freedom
## Multiple R-squared:  0.4263, Adjusted R-squared:  0.1446
## F-statistic: 1.513 on 55 and 112 DF,  p-value: 0.03312
```

```
AIC(lin.mod5.full)
```

```
## [1] 1002.049
```

```
yhat <- predict(lin.mod5.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod5.full, newdata = testing): prediction from a rank-
## deficient fit may be misleading
```

```
y.test <- testing[, "relatDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 23.63526
```

```
lin.mod6.full <- lm(clasfDiff ~ (site + sex + age + viewcat + setting + viewenc)^2, data = training)
summary(lin.mod6.full)
```

```
##
## Call:
## lm(formula = clasfDiff ~ (site + sex + age + viewcat + setting +
##   viewenc)^2, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.6108  -3.1873   0.0668   3.0854  10.7635
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    31.86932   20.13470   1.583  0.1163
## site2           5.71113   18.87658   0.303  0.7628
## site3          -11.78957   12.56551  -0.938  0.3501
## site4          -35.64786   22.68842  -1.571  0.1190
## site5          -56.84008   23.94777  -2.374  0.0193 *
## sex2           16.72781    9.35194   1.789  0.0764 .
## age            -0.37531    0.36829  -1.019  0.3104
## viewcat2       -33.62976   20.33787  -1.654  0.1010
## viewcat3       -33.98623   18.38467  -1.849  0.0672 .
## viewcat4       -12.80847   20.00391  -0.640  0.5233
## setting2        25.39476   12.54023   2.025  0.0452 *
## viewenc2        -9.74311   14.04214  -0.694  0.4892
## site2:sex2      -0.52039    3.06238  -0.170  0.8654
## site3:sex2       2.53032    2.98486   0.848  0.3984
## site4:sex2      -3.37095    3.62448  -0.930  0.3543
## site5:sex2      -4.87172    5.11219  -0.953  0.3427
## site2:age        0.08205    0.33065   0.248  0.8045
## site3:age        0.16320    0.23587   0.692  0.4904
## site4:age        0.80312    0.42105   1.907  0.0590 .
## site5:age        1.15008    0.46355   2.481  0.0146 *
## site2:viewcat2  -2.93114    7.57584  -0.387  0.6996
## site3:viewcat2   5.06294    5.34770   0.947  0.3458
## site4:viewcat2   3.68884    5.36739   0.687  0.4933
## site5:viewcat2   5.58665    8.64949   0.646  0.5197
## site2:viewcat3  -8.51958    7.09289  -1.201  0.2322
## site3:viewcat3   0.53577    5.04213   0.106  0.9156
## site4:viewcat3   1.94832    5.71524   0.341  0.7338
## site5:viewcat3      NA         NA       NA      NA
## site2:viewcat4 -11.97418    7.20383  -1.662  0.0993 .
## site3:viewcat4   6.67420    5.28661   1.262  0.2094
## site4:viewcat4  -7.87519    6.68380  -1.178  0.2412
## site5:viewcat4  -2.47008    9.12901  -0.271  0.7872
## site2:setting2   3.68492    3.80212   0.969  0.3345
## site3:setting2  -6.53049    4.35461  -1.500  0.1365
## site4:setting2  -3.60393    4.47448  -0.805  0.4223
## site5:setting2      NA         NA       NA      NA
## site2:viewenc2  -3.02265    3.58254  -0.844  0.4006
## site3:viewenc2   3.51672    4.54535   0.774  0.4407
```

```
## site4:viewenc2      1.12474      4.17152      0.270      0.7879
## site5:viewenc2     12.00787      9.75463      1.231      0.2209
## sex2:age           -0.28474      0.17403     -1.636      0.1046
## sex2:viewcat2       1.58139      3.64388      0.434      0.6651
## sex2:viewcat3       1.22311      3.45301      0.354      0.7238
## sex2:viewcat4      -3.07878      3.68935     -0.835      0.4058
## sex2:setting2      -3.66101      2.34297     -1.563      0.1210
## sex2:viewenc2       1.89995      2.66823      0.712      0.4779
## age:viewcat2        0.59861      0.36006      1.663      0.0992 .
## age:viewcat3        0.67510      0.32533      2.075      0.0403 *
## age:viewcat4        0.32333      0.35835      0.902      0.3688
## age:setting2       -0.32330      0.21177     -1.527      0.1297
## age:viewenc2       -0.03592      0.24933     -0.144      0.8857
## viewcat2:setting2  -9.02471      4.55447     -1.982      0.0500 *
## viewcat3:setting2  -7.57750      4.40886     -1.719      0.0884 .
## viewcat4:setting2  -1.82366      4.85327     -0.376      0.7078
## viewcat2:viewenc2  10.57132      4.87489      2.169      0.0322 *
## viewcat3:viewenc2  11.25111      4.60050      2.446      0.0160 *
## viewcat4:viewenc2   6.93655      4.98500      1.391      0.1668
## setting2:viewenc2   1.22748      3.19174      0.385      0.7013
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.813 on 112 degrees of freedom
## Multiple R-squared:  0.4256, Adjusted R-squared:  0.1436
## F-statistic: 1.509 on 55 and 112 DF,  p-value: 0.03404
```

```
AIC(lin.mod6.full)
```

```
## [1] 1114.029
```

```
yhat <- predict(lin.mod6.full, newdata = testing)
```

```
## Warning in predict.lm(lin.mod6.full, newdata = testing): prediction from a rank-
## deficient fit may be misleading
```

```
y.test <- testing[, "clasfDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 65.41
```

Regression Tree Models

In the context of a regression tree, the deviance is simply the sum of the squared errors.

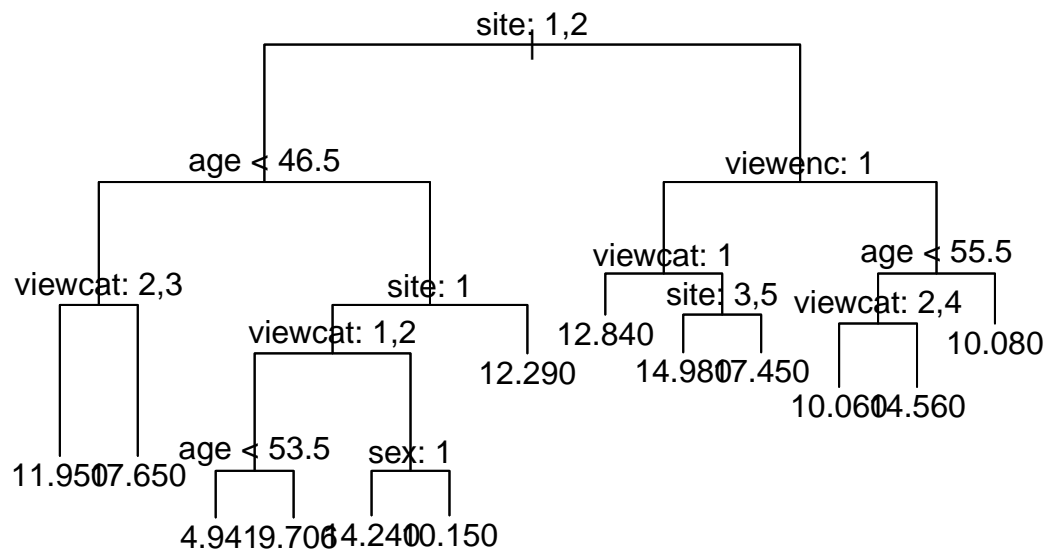
Model 1

```
set.seed(1)
```

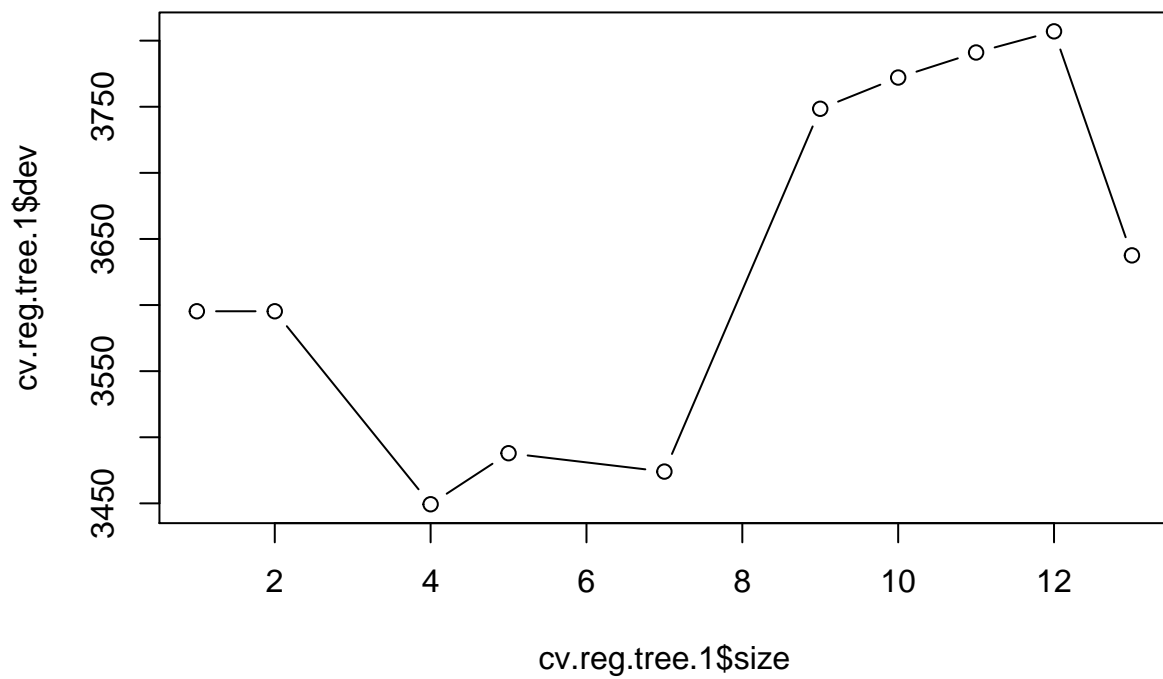
```
reg.tree.1 <- tree(bodyDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.1)
```

```
##
## Regression tree:
## tree(formula = bodyDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Variables actually used in tree construction:
## [1] "site"    "age"     "viewcat" "sex"     "viewenc"
## Number of terminal nodes: 13
## Residual mean deviance: 14.16 = 2194 / 155
## Distribution of residuals:
##      Min.   1st Qu.   Median     Mean   3rd Qu.    Max.
## -11.08000  -2.45100  -0.06303   0.00000   2.16900  12.55000
```

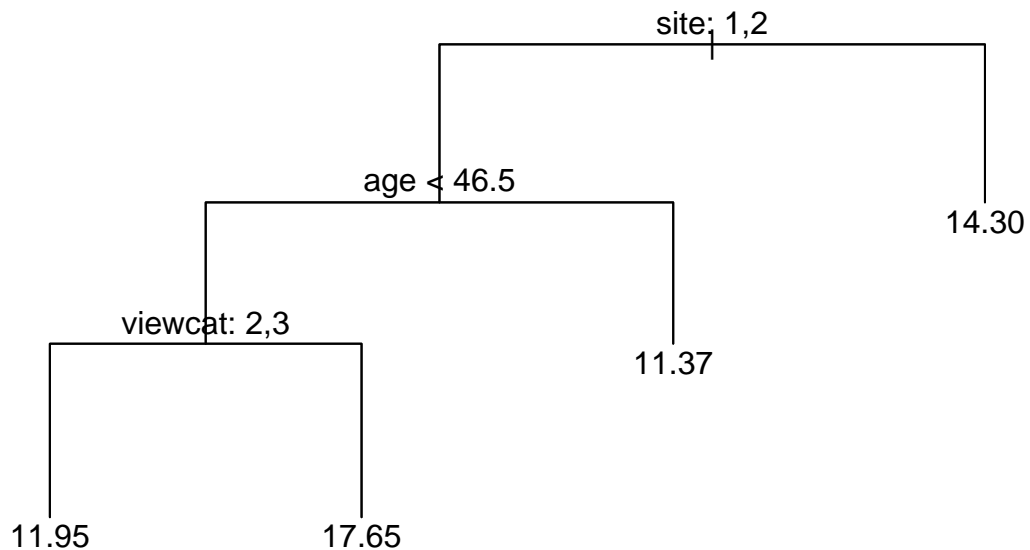
```
plot(reg.tree.1)
text(reg.tree.1, pretty = 0)
```



```
cv.reg.tree.1 <- cv.tree(reg.tree.1)
plot(cv.reg.tree.1$size, cv.reg.tree.1$dev, type = "b")
```



```
prune.reg.tree.1 <- prune.tree(reg.tree.1, best = 4)
plot(prune.reg.tree.1)
text(prune.reg.tree.1, pretty = 0)
```



```
yhat <- predict(prune.reg.tree.1, newdata = testing)
y.test <- testing[, "bodyDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 20.60159
```

Model 2

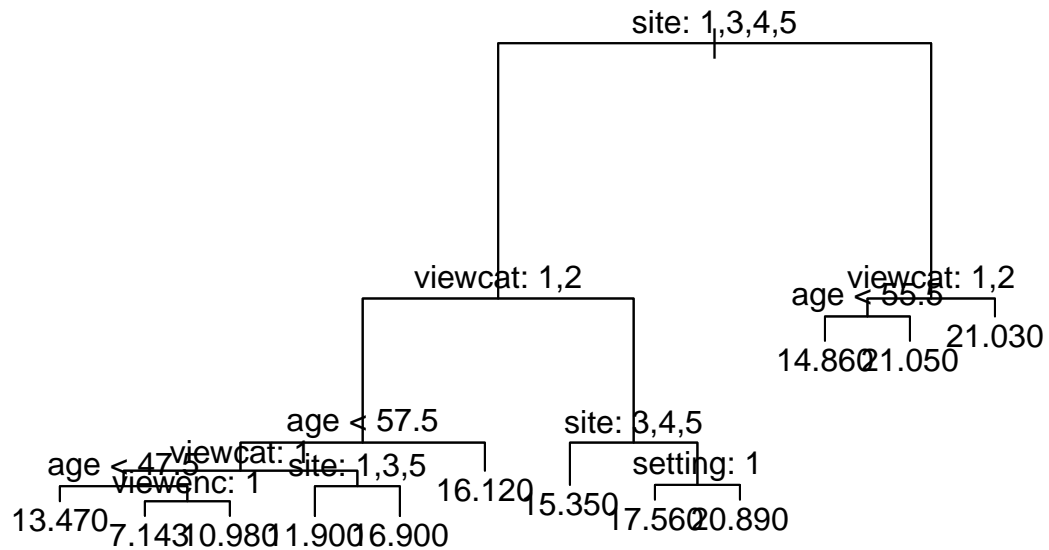
```
set.seed(1)
```

```
reg.tree.2 <- tree(letDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.2)
```

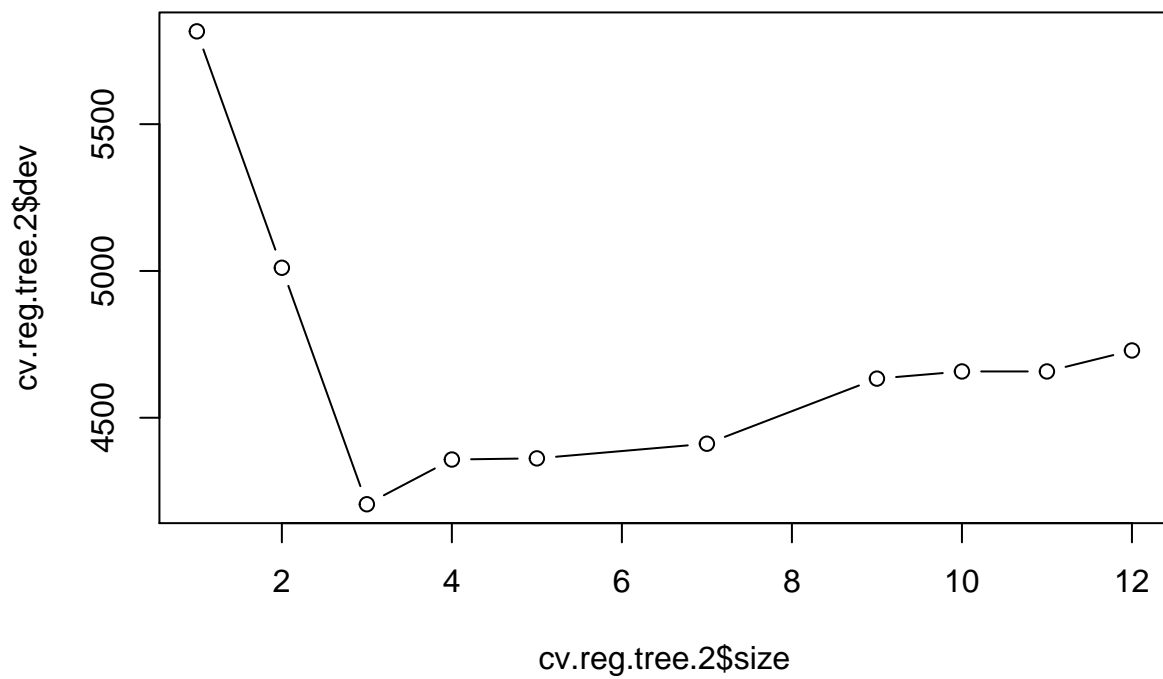
```
##
## Regression tree:
## tree(formula = letDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Variables actually used in tree construction:
## [1] "site"      "viewcat"   "age"       "viewenc"   "setting"
## Number of terminal nodes: 12
## Residual mean deviance: 18.63 = 2906 / 156
```

```
## Distribution of residuals:
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -17.560 -2.430   0.000   0.000   2.765   9.587
```

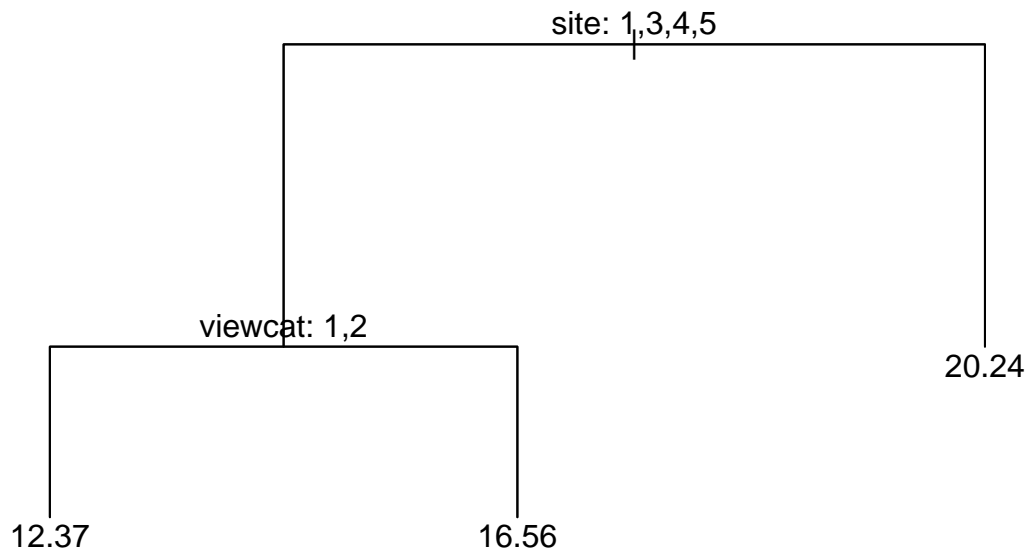
```
plot(reg.tree.2)
text(reg.tree.2, pretty = 0)
```



```
cv.reg.tree.2 <- cv.tree(reg.tree.2)
plot(cv.reg.tree.2$size, cv.reg.tree.2$dev, type = "b")
```

```
prune.reg.tree.2 <- prune.tree(reg.tree.2, best = 3)
plot(prune.reg.tree.2)
text(prune.reg.tree.2, pretty = 0)
```



```

yhat <- predict(prune.reg.tree.2, newdata = testing)
y.test <- testing[, "letDiff"]

# Test MSE
mean((yhat-y.test)^2)

```

```
## [1] 15.4124
```

Model 3

```

set.seed(1)

reg.tree.3 <- tree(formDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.3)

```

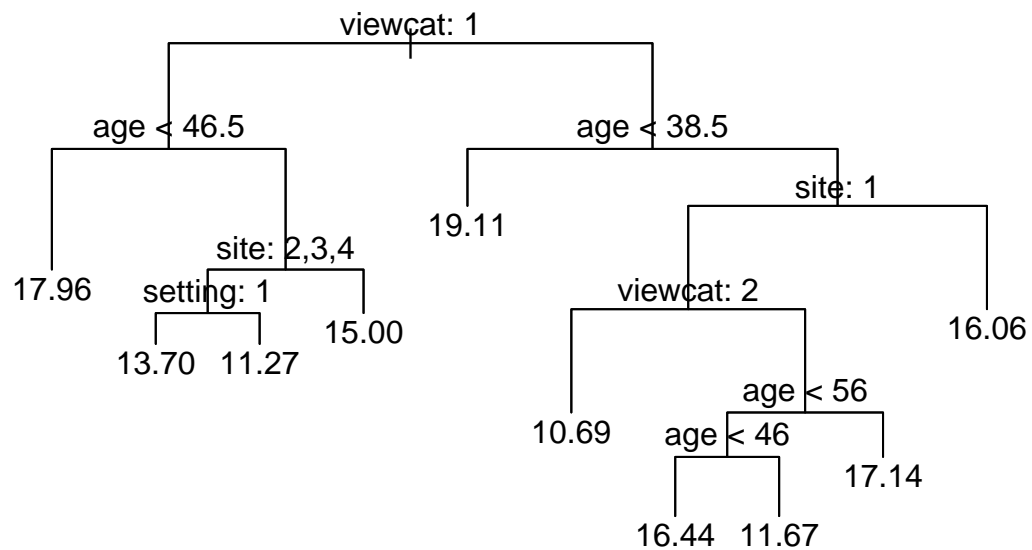
```

##
## Regression tree:
## tree(formula = formDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Variables actually used in tree construction:
## [1] "viewcat" "age"      "site"      "setting"
## Number of terminal nodes: 10
## Residual mean deviance: 15.88 = 2509 / 158

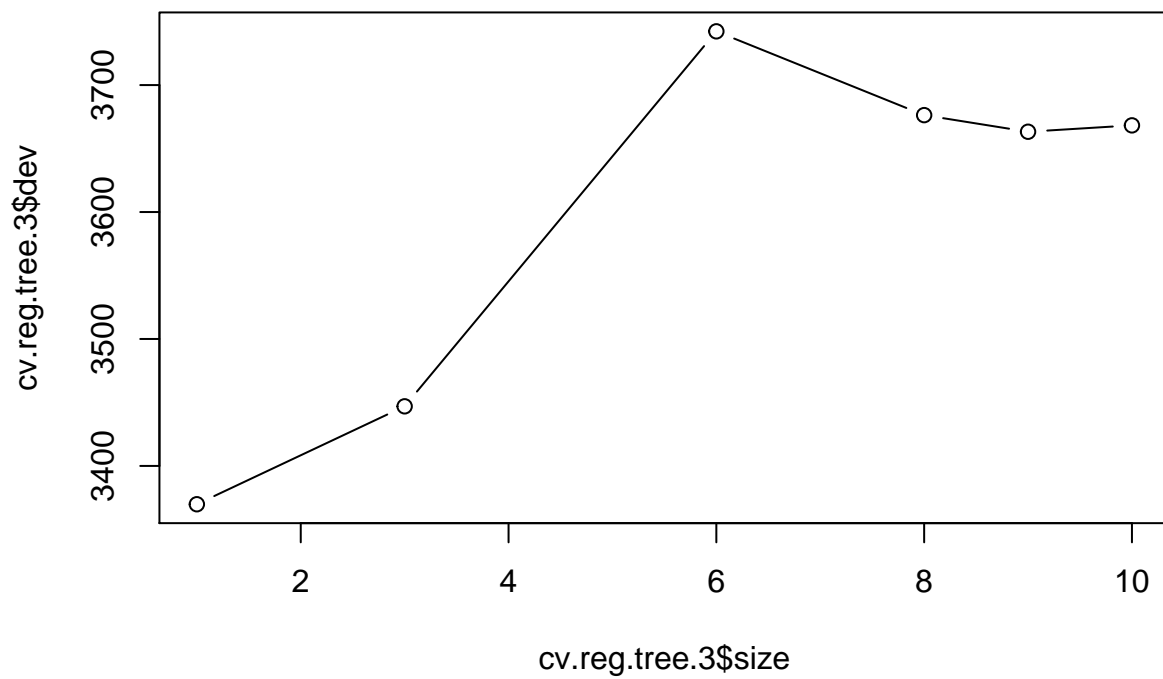
```

```
## Distribution of residuals:
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -11.610 -1.667   0.129   0.000   2.159   13.940
```

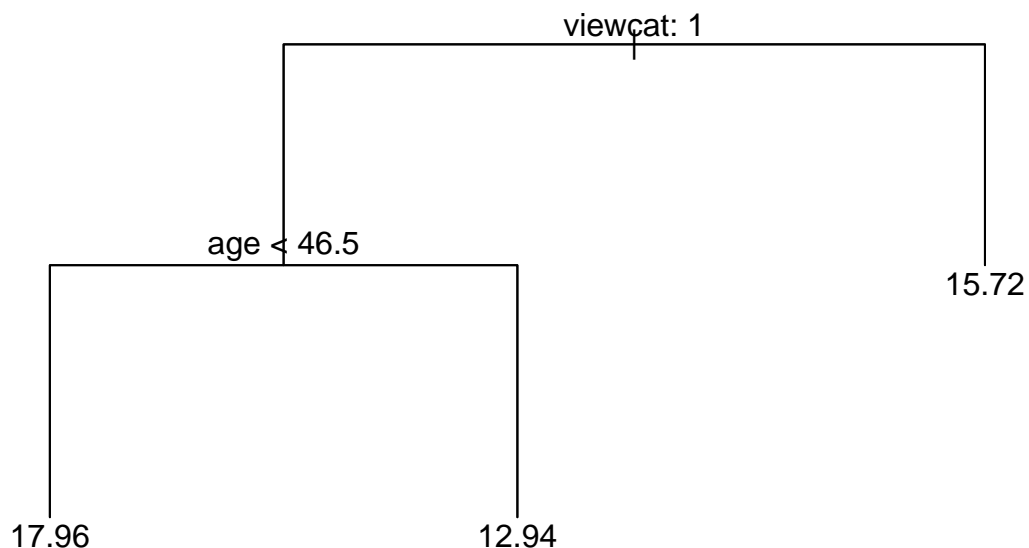
```
plot(reg.tree.3)
text(reg.tree.3, pretty = 0)
```



```
cv.reg.tree.3 <- cv.tree(reg.tree.3)
plot(cv.reg.tree.3$size, cv.reg.tree.3$dev, type = "b")
```



```
prune.reg.tree.3 <- prune.tree(reg.tree.3, best = 2)
plot(prune.reg.tree.3)
text(prune.reg.tree.3, pretty = 0)
```



```
yhat <- predict(prune.reg.tree.3, newdata = testing)
y.test <- testing[, "formDiff"]
```

```
# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 14.92273
```

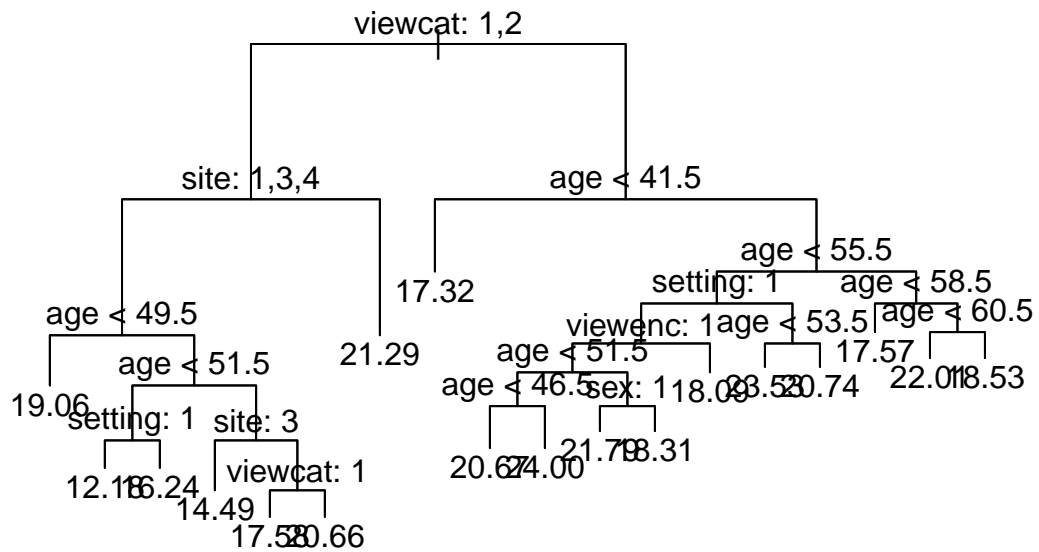
Model 4

```
set.seed(1)
```

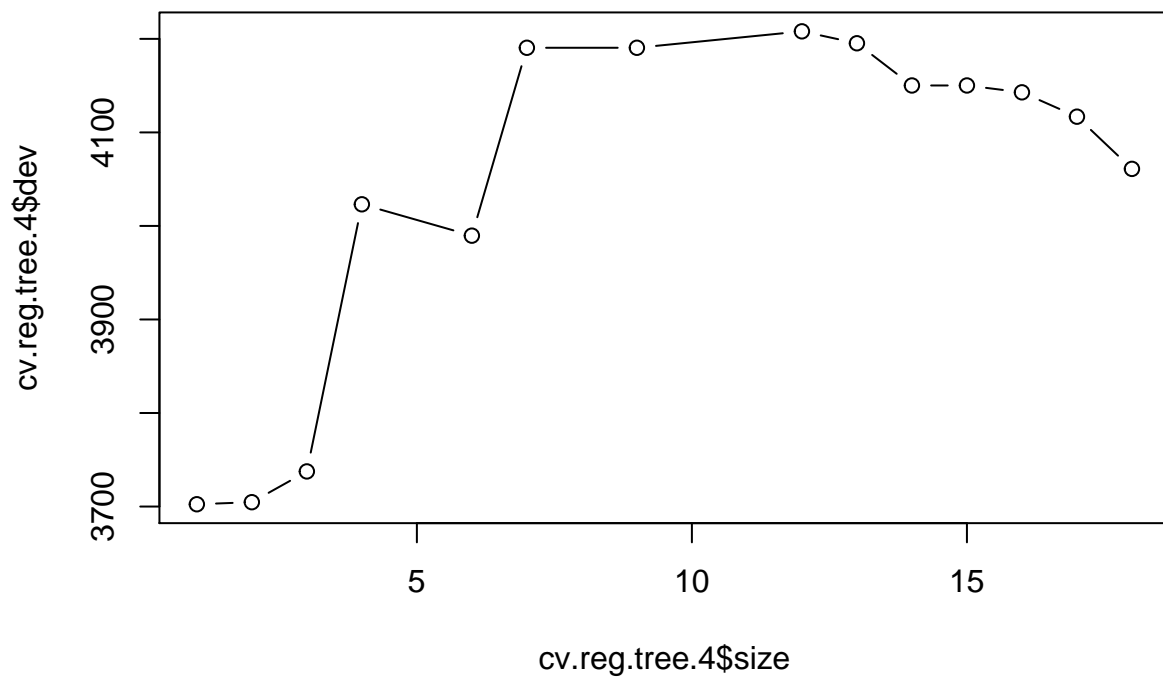
```
reg.tree.4 <- tree(numDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.4)
```

```
##
## Regression tree:
## tree(formula = numDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Number of terminal nodes: 18
## Residual mean deviance: 13.64 = 2046 / 150
## Distribution of residuals:
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -18.0900 -2.1210   0.2647   0.0000  2.1180  11.4700
```

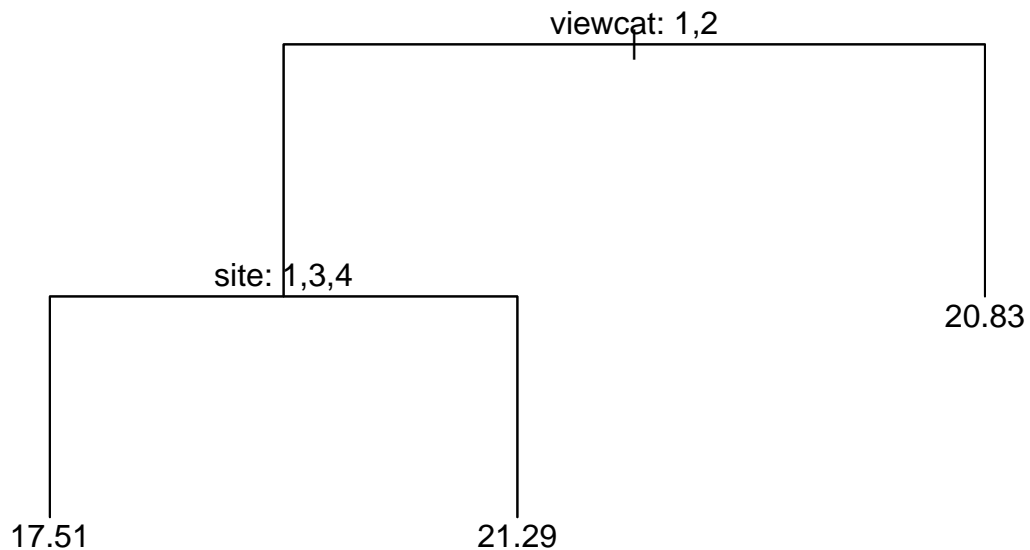
```
plot(reg.tree.4)
text(reg.tree.4, pretty = 0)
```



```
cv.reg.tree.4 <- cv.tree(reg.tree.4)
plot(cv.reg.tree.4$size, cv.reg.tree.4$dev, type = "b")
```



```
prune.reg.tree.4 <- prune.tree(reg.tree.4, best = 3)
plot(prune.reg.tree.4)
text(prune.reg.tree.4, pretty = 0)
```



```

yhat <- predict(prune.reg.tree.4, newdata = testing)
y.test <- testing[, "numbDiff"]

# Test MSE
mean((yhat-y.test)^2)

```

```
## [1] 15.90771
```

Model 5

```

set.seed(1)

reg.tree.5 <- tree(relatDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.5)

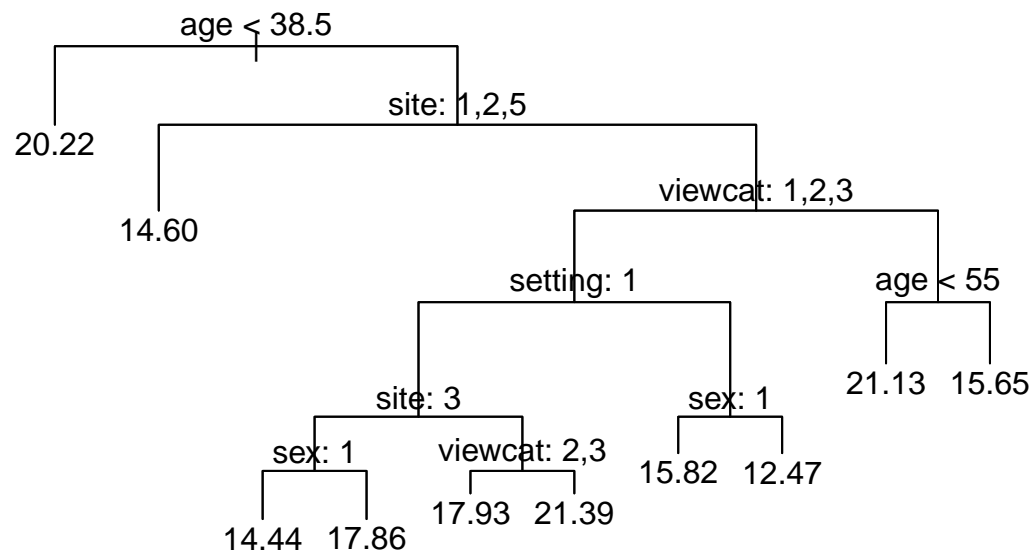
##
## Regression tree:
## tree(formula = relatDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Variables actually used in tree construction:
## [1] "age"      "site"     "viewcat"  "setting"  "sex"
## Number of terminal nodes: 10
## Residual mean deviance: 15.3 = 2418 / 158

```

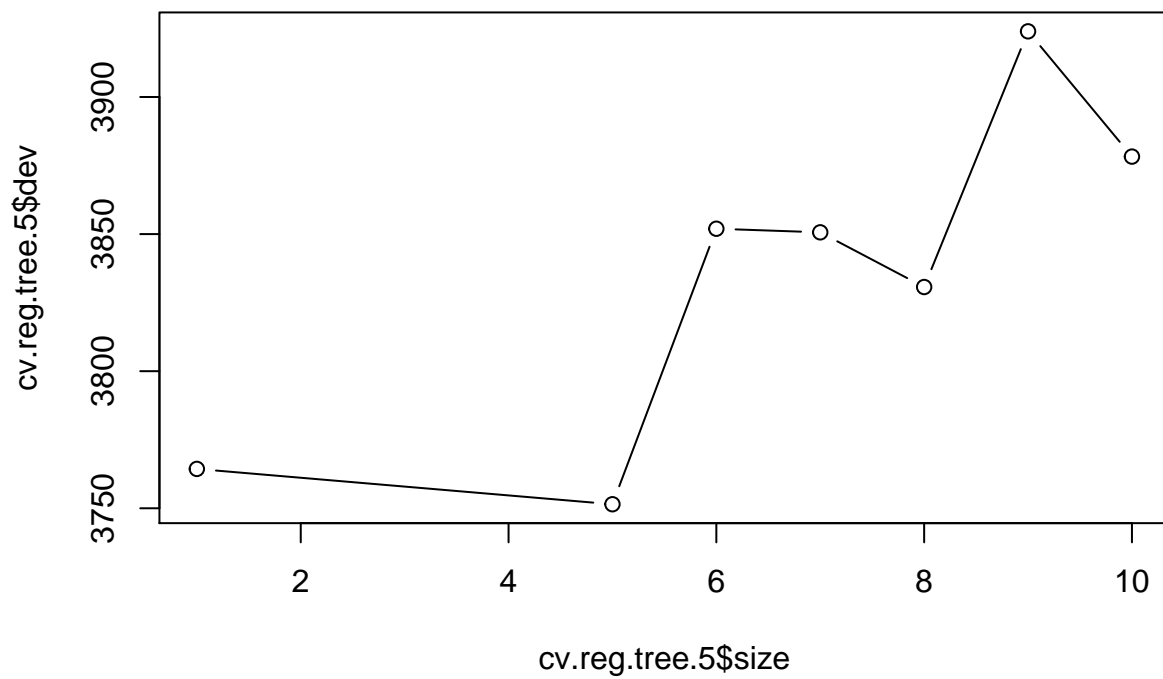


```
## Distribution of residuals:
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -10.6900 -1.5650 -0.2514  0.0000  1.8950  11.4900
```

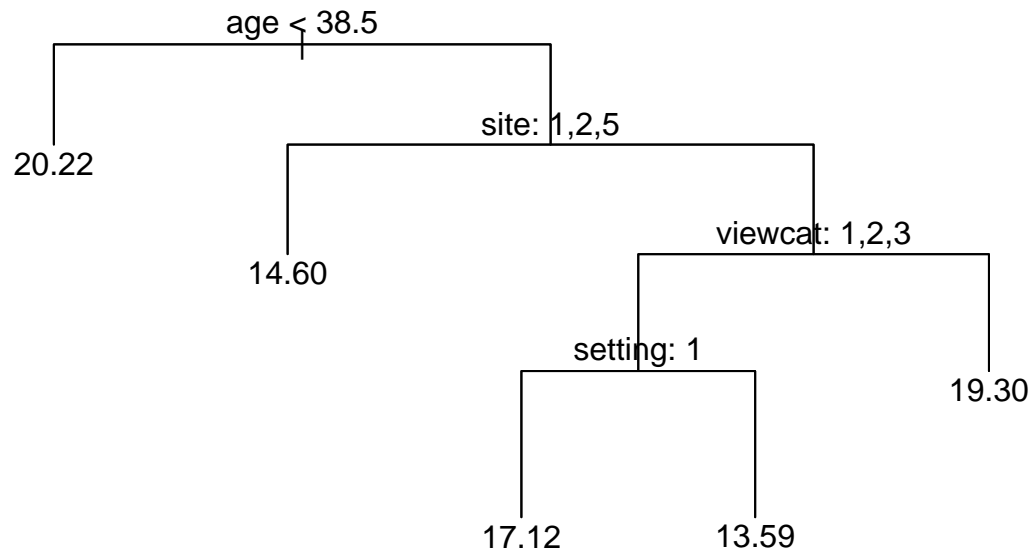
```
plot(reg.tree.5)
text(reg.tree.5, pretty = 0)
```



```
cv.reg.tree.5 <- cv.tree(reg.tree.5)
plot(cv.reg.tree.5$size, cv.reg.tree.5$dev, type = "b")
```



```
prune.reg.tree.5 <- prune.tree(reg.tree.5, best = 5)
plot(prune.reg.tree.5)
text(prune.reg.tree.5, pretty = 0)
```



```
yhat <- predict(prune.reg.tree.5, newdata = testing)
y.test <- testing[, "relatDiff"]

# Test MSE
mean((yhat-y.test)^2)
```

```
## [1] 19.88506
```

Model 6

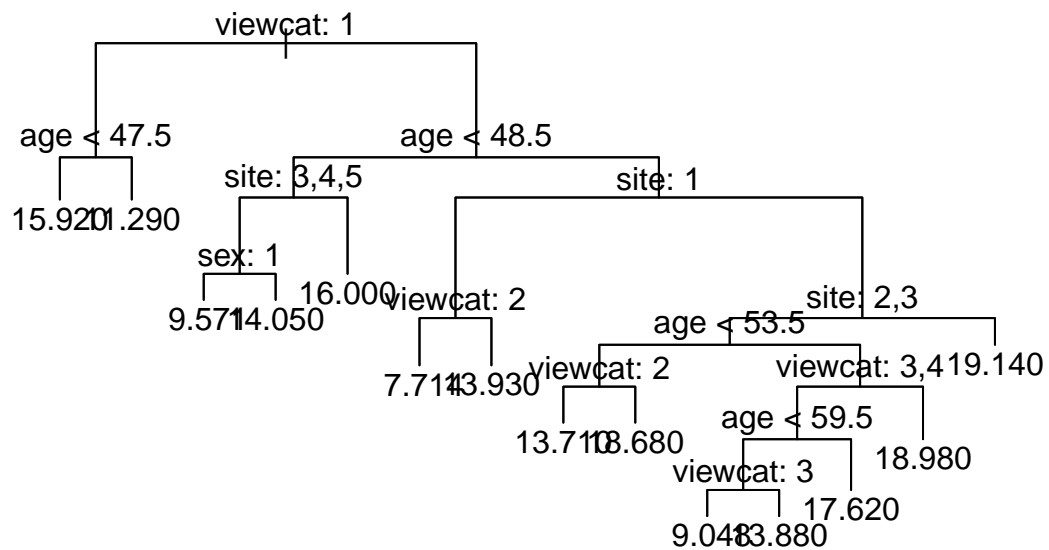
```
set.seed(1)

reg.tree.6 <- tree(clasfDiff ~ site + sex + age + viewcat + setting + viewenc, sesame.q1, subset = train)
summary(reg.tree.6)

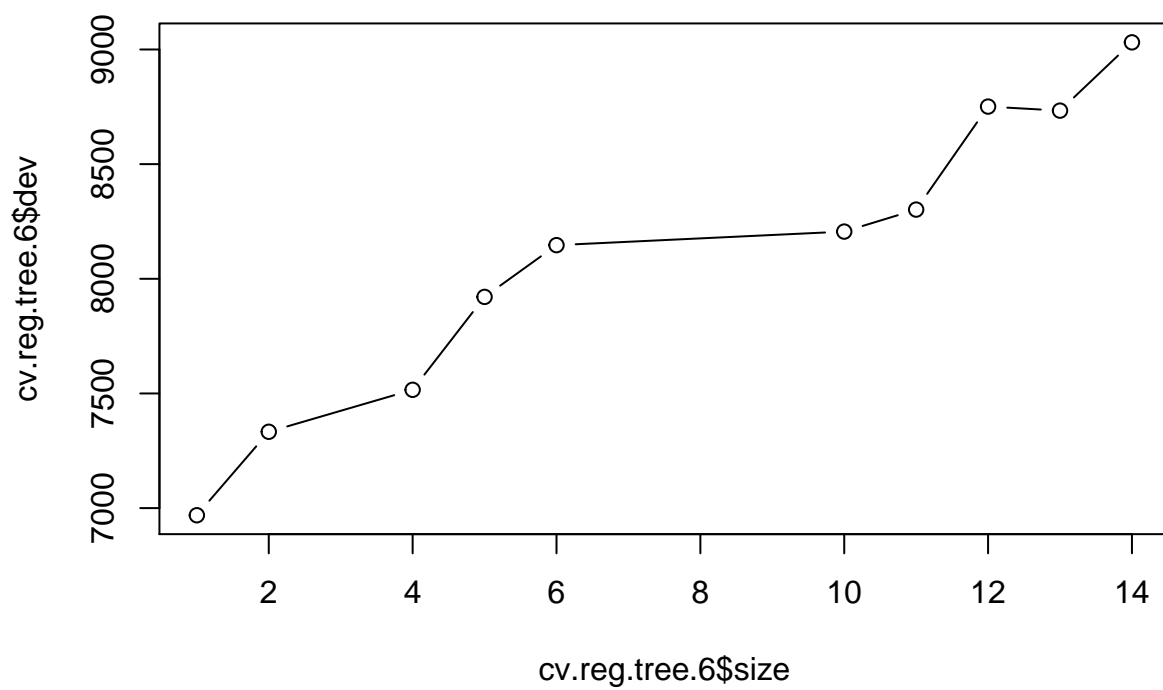
##
## Regression tree:
## tree(formula = clasfDiff ~ site + sex + age + viewcat + setting +
##       viewenc, data = sesame.q1, subset = train)
## Variables actually used in tree construction:
## [1] "viewcat" "age"      "site"     "sex"
## Number of terminal nodes: 14
## Residual mean deviance: 29.66 = 4568 / 154
```

```
## Distribution of residuals:
##      Min.   1st Qu.   Median     Mean   3rd Qu.    Max.
## -16.00000  -3.17300   0.01852   0.00000   3.71400   13.10000
```

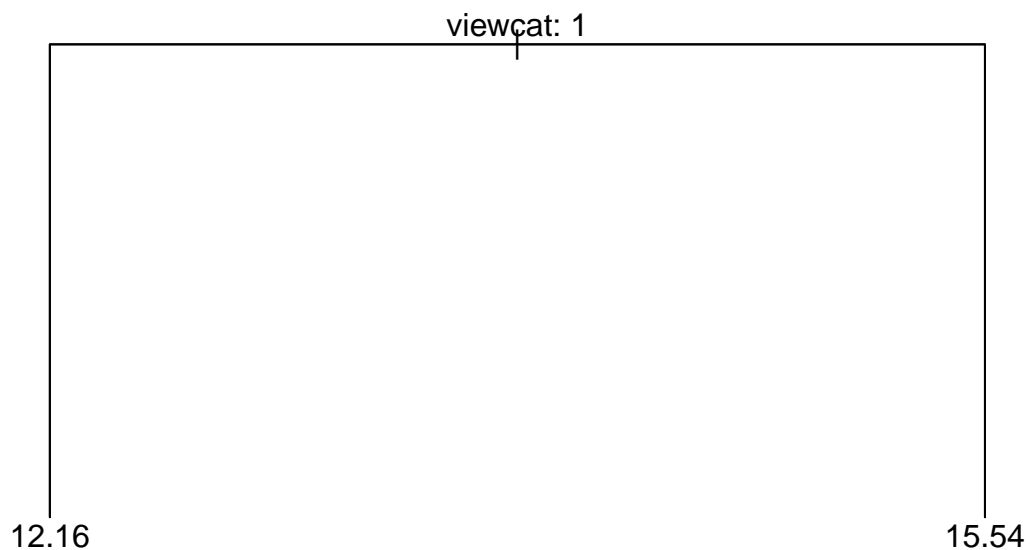
```
plot(reg.tree.6)
text(reg.tree.6, pretty = 0)
```



```
cv.reg.tree.6 <- cv.tree(reg.tree.6)
plot(cv.reg.tree.6$size, cv.reg.tree.6$dev, type = "b")
```



```
prune.reg.tree.6 <- prune.tree(reg.tree.6, best = 2)
plot(prune.reg.tree.6)
text(prune.reg.tree.6, pretty = 0)
```



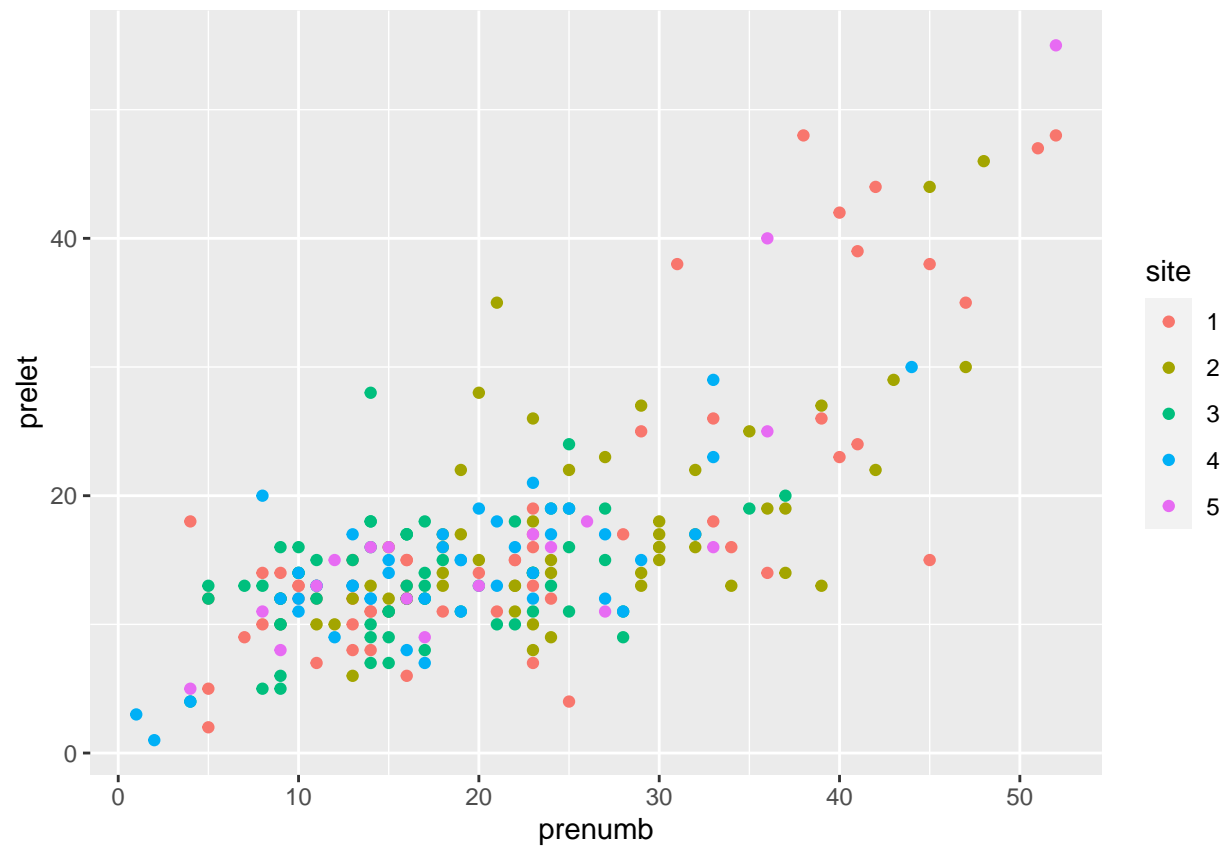
```
yhat <- predict(prune.reg.tree.6, newdata = testing)
y.test <- testing[, "clasfDiff"]

# Test MSE
mean((yhat-y.test)^2)
```

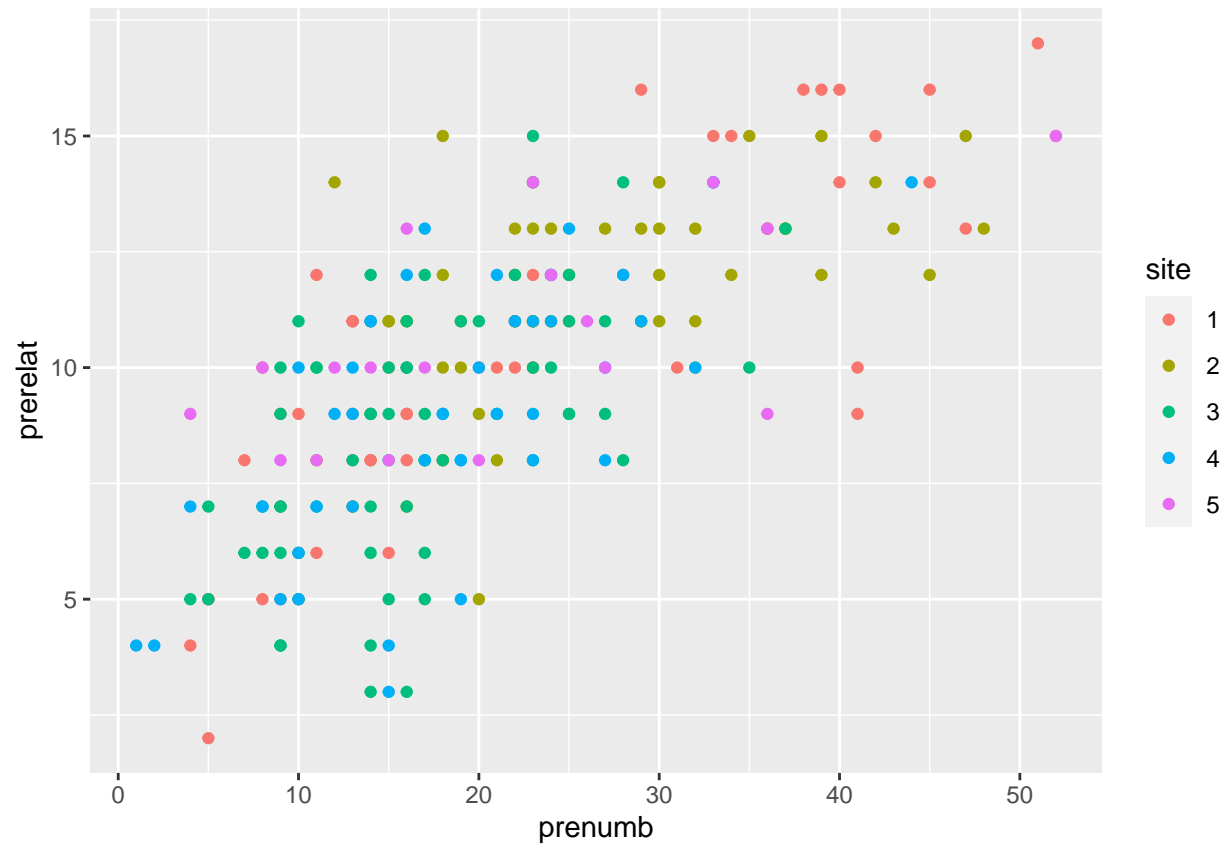
```
## [1] 45.52784
```

Q.2 Classification Question: Can we use the pre-test scores and other demographic variables to predict which region the children came from?

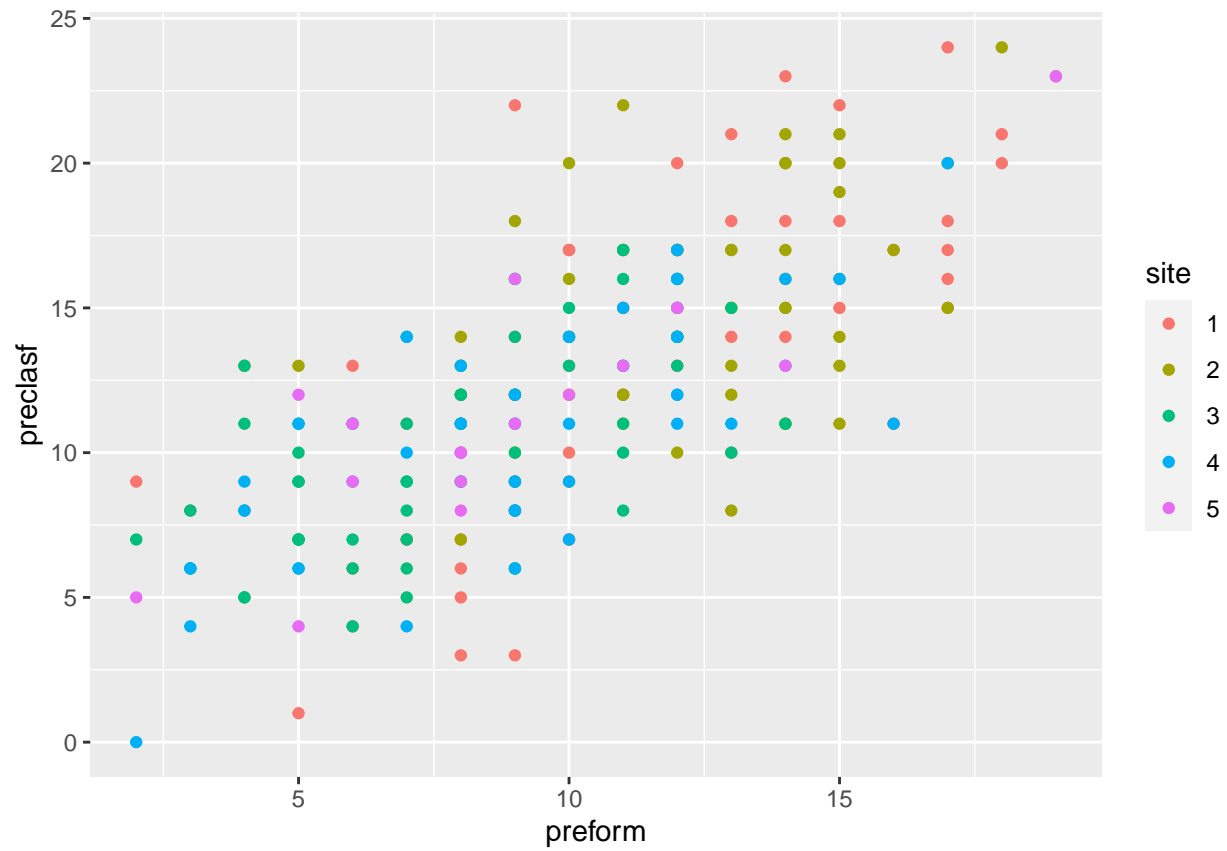
```
ggplot(data=sesame, aes(x=prenumb, y=prelet, color=site))+
  geom_point()
```



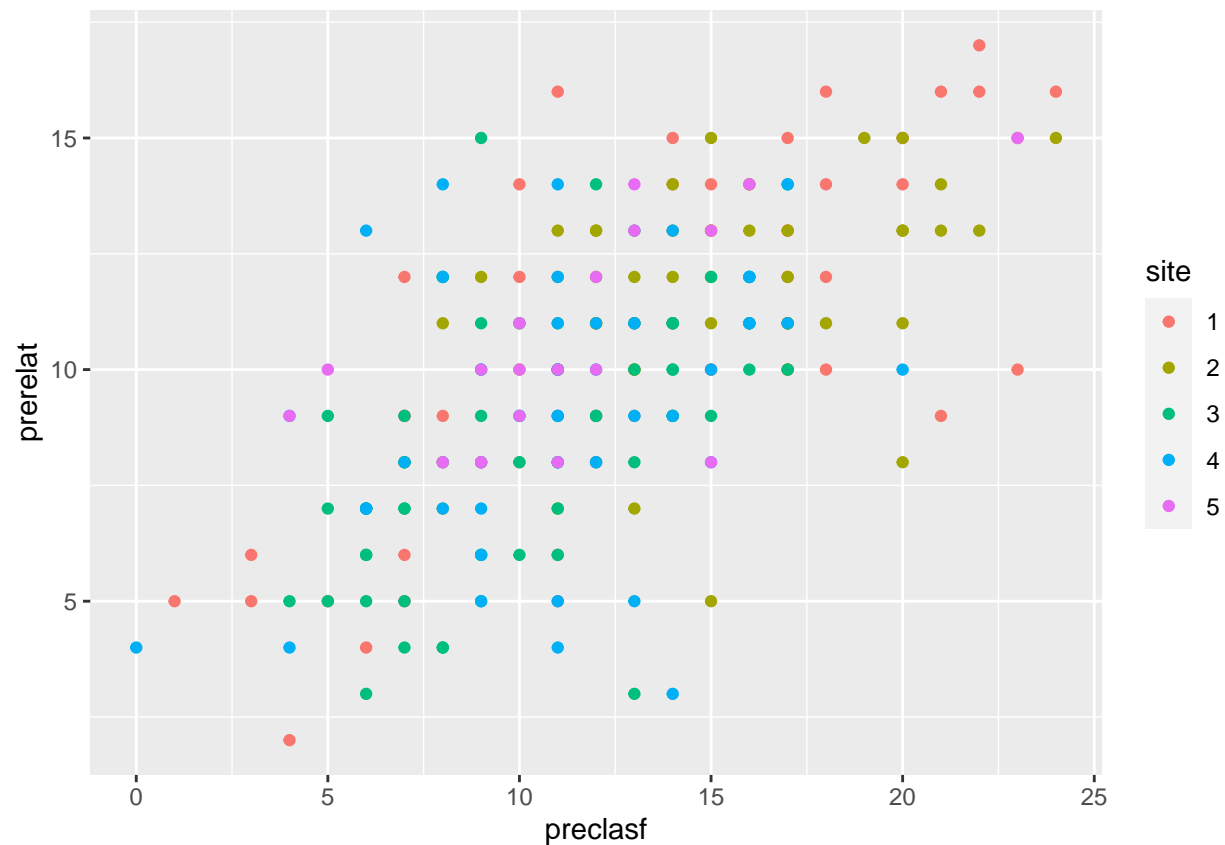
```
ggplot(data=sesame, aes(x=prenumb, y=prerelat, color=site))+  
  geom_point()
```



```
ggplot(data=sesame, aes(x=preform, y=preclasf, color=site))+
  geom_point()
```

```
ggplot(data=sesame, aes(x=preclasf, y=prerelat, color=site))+
  geom_point()
```



SVM

```
set.seed(3241)

n <- nrow(sesame)
train.index <- sample(1:n, size = floor(0.7*n), replace=FALSE)
train.data <- sesame.sd[train.index,]
test.data <- sesame.sd[-train.index,]

train.data %>%
  count(site)
```

```
##   site  n
## 1     1 40
## 2     2 42
## 3     3 48
## 4     4 25
## 5     5 13
```

```
#1 60
#2 55
#3 64
#4 43
```

```
#5 18
```

```
total.weight <- 60+55+64+43+18
weight.1 <- total.weight/(5*60)
weight.2 <- total.weight/(5*55)
weight.3 <- total.weight/(5*64)
weight.4 <- total.weight/(5*43)
weight.5 <- total.weight/(5*18)
```

```
weight.4 <- 1.5
weight.5 <- 3
```

```
# Response: site (categorical)
```

```
set.seed(315)
```

```
costs <- c(0.001, 0.01, 0.1, 1, 5, 10, 100)
```

```
# c(0.1, 0.2, 0.5, 0.7, 1, 2, 3, 4)
```

```
gammas <- seq(0, 4, by=0.1)
```

```
linear.tune <- tune(svm, site~female+ male + sd_age+sd_pBod+sd_plet+sd_pform + sd_pnumb+sd_prelat+sd_pclasf,
                   data=train.data, kernel="linear",
                   ranges=list(cost=costs),
                   class.weights=c("1"=weight.1,
                                   "2"=weight.2,
                                   "3"=weight.3,
                                   "4"=weight.4,
                                   "5"=weight.5),
                   class.type="one.versus.one")
```

```
radial.tune <- tune(svm, site~female + male + sd_age+sd_pBod+sd_plet+sd_pform + sd_pnumb+sd_prelat+sd_pclasf,
                   data=train.data, kernel="radial",
                   ranges=list(cost=costs,
                               gamma=gammas),
                   class.weights=c("1"=weight.1,
                                   "2"=weight.2,
                                   "3"=weight.3,
                                   "4"=weight.4,
                                   "5"=weight.5))
```

```
#radial.tune <- tune(svm, site~sex+age+prebody+prelet+preform+prenumb+prerelat+preclasf,
#                     data=train.data, kernel="radial",
#                     ranges=list(cost=costs,
#                                 gamma=gammas))
```

```
sigmoid.tune <- tune(svm, site~female + male + sd_age+sd_pBod+sd_plet+sd_pform + sd_pnumb+sd_prelat+sd_pclasf,
                    data=train.data, kernel="sigmoid",
                    ranges=list(cost=costs,
                                gamma=gammas),
                    class.weights=c("1"=weight.1,
                                    "2"=weight.2,
                                    "3"=weight.3,
                                    "4"=weight.4,
                                    "5"=weight.5))
```

```
linear.conMatrix <- table(true=test.data[, "site"],
                          pred=predict(linear.tune$best.model, newdata=test.data))

radial.conMatrix <- table(true=test.data[, "site"],
                          pred=predict(radial.tune$best.model, newdata=test.data))

sigmoid.conMatrix <- table(true=test.data[, "site"],
                           pred=predict(sigmoid.tune$best.model, newdata=test.data))

confusionMatrix(linear.conMatrix)
```

```
## Confusion Matrix and Statistics
##
##      pred
## true  1  2  3  4  5
##      1  6  1  6  2  5
##      2  1  7  1  1  3
##      3  0  1 11  1  3
##      4  3  3  9  3  0
##      5  0  1  1  1  2
##
## Overall Statistics
##
##              Accuracy : 0.4028
##              95% CI : (0.2888, 0.525)
##      No Information Rate : 0.3889
##      P-Value [Acc > NIR] : 0.44844
##
##              Kappa : 0.2554
##
##  McNemar's Test P-Value : 0.01728
##
## Statistics by Class:
##
##              Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity          0.60000  0.53846  0.3929  0.37500  0.15385
## Specificity          0.77419  0.89831  0.8864  0.76562  0.94915
## Pos Pred Value       0.30000  0.53846  0.6875  0.16667  0.40000
## Neg Pred Value       0.92308  0.89831  0.6964  0.90741  0.83582
## Prevalence           0.13889  0.18056  0.3889  0.11111  0.18056
## Detection Rate       0.08333  0.09722  0.1528  0.04167  0.02778
## Detection Prevalence 0.27778  0.18056  0.2222  0.25000  0.06944
## Balanced Accuracy    0.68710  0.71838  0.6396  0.57031  0.55150
```

```
confusionMatrix(radial.conMatrix)
```

```
## Confusion Matrix and Statistics
##
##      pred
## true  1  2  3  4  5
##      1  6  2  7  4  1
##      2  4  4  3  0  2
##      3  1  1 11  3  0
```

```
##      4  2  2  9  4  1
##      5  0  2  1  1  1
##
## Overall Statistics
##
##              Accuracy : 0.3611
##              95% CI : (0.2512, 0.4829)
##      No Information Rate : 0.4306
##      P-Value [Acc > NIR] : 0.9056
##
##              Kappa : 0.181
##
## McNemar's Test P-Value : 0.1807
##
## Statistics by Class:
##
##              Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.46154  0.36364  0.3548  0.33333  0.20000
## Specificity      0.76271  0.85246  0.8780  0.76667  0.94030
## Pos Pred Value   0.30000  0.30769  0.6875  0.22222  0.20000
## Neg Pred Value   0.86538  0.88136  0.6429  0.85185  0.94030
## Prevalence       0.18056  0.15278  0.4306  0.16667  0.06944
## Detection Rate   0.08333  0.05556  0.1528  0.05556  0.01389
## Detection Prevalence 0.27778  0.18056  0.2222  0.25000  0.06944
## Balanced Accuracy 0.61213  0.60805  0.6164  0.55000  0.57015
```

```
confusionMatrix(sigmoid.conMatrix)
```

```
## Confusion Matrix and Statistics
##
##      pred
## true  1  2  3  4  5
##      1  1  1  6  9  3
##      2  1  3  4  5  0
##      3  1  0  4 11  0
##      4  0  0  7 11  0
##      5  0  0  3  2  0
##
## Overall Statistics
##
##              Accuracy : 0.2639
##              95% CI : (0.167, 0.381)
##      No Information Rate : 0.5278
##      P-Value [Acc > NIR] : 1
##
##              Kappa : 0.0434
##
## McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##              Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.33333  0.75000  0.16667  0.2895  0.00000
## Specificity      0.72464  0.85294  0.75000  0.7941  0.92754
```

```
predict(linear.tune$best.model, newdata=test.data)
```

```
predict(radial.tune$best.model, newdata=test.data)
```

```
predict(sigmoid.tune$best.model, newdata=test.data)
```

```
test.data$site
```

46

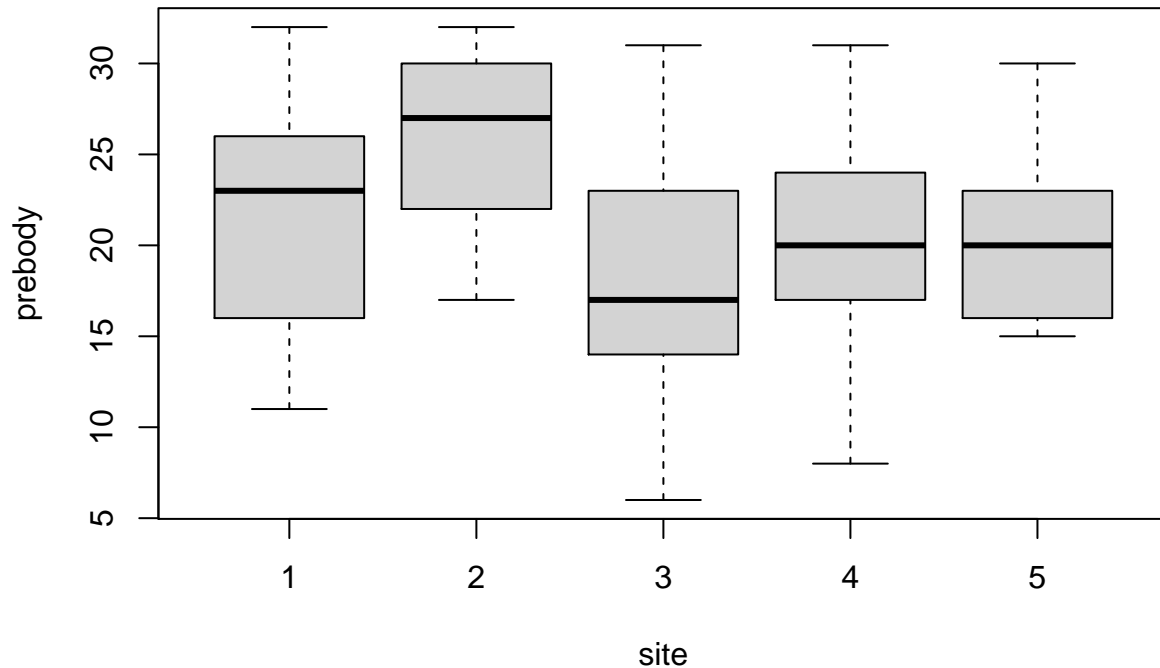
RBF slightly improved after standardizing? (it seems slightly more likely to predict on class 1.) thought, simpler models still retain the same performance (arguably better) `sd_age+sd_pBod+sd_plet`. But we are still not getting any prediction on class 4 & 5.

After assign class weights using this formula:

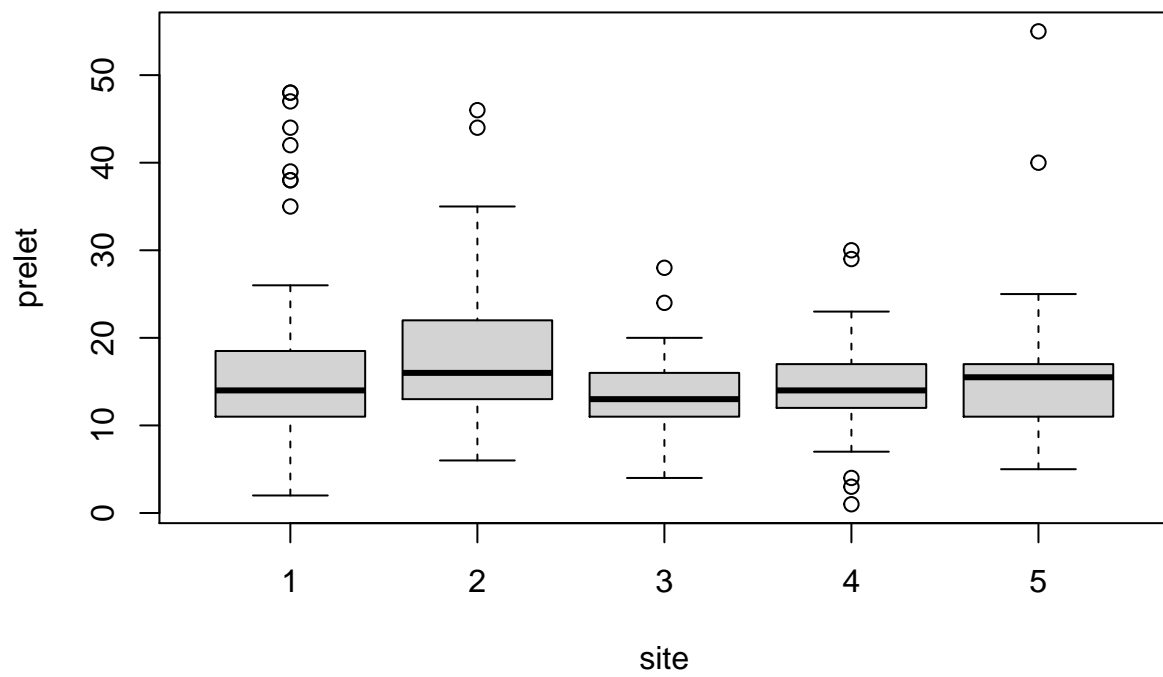
$$w_j = \frac{n}{kn_j}, \text{ n is total number of data points, k is number of classes}$$

Our model begins to make predictions on class 4 & class 5, though at the cost of overall accuracy. If we increase the weight for 4 & 5 to 1.5 and 3 respectively the performance of Radial SVM decreases but that of linear SVM increases to be comparable to Radial SVM's recorded highest accuracy (a little bit over 0.40).

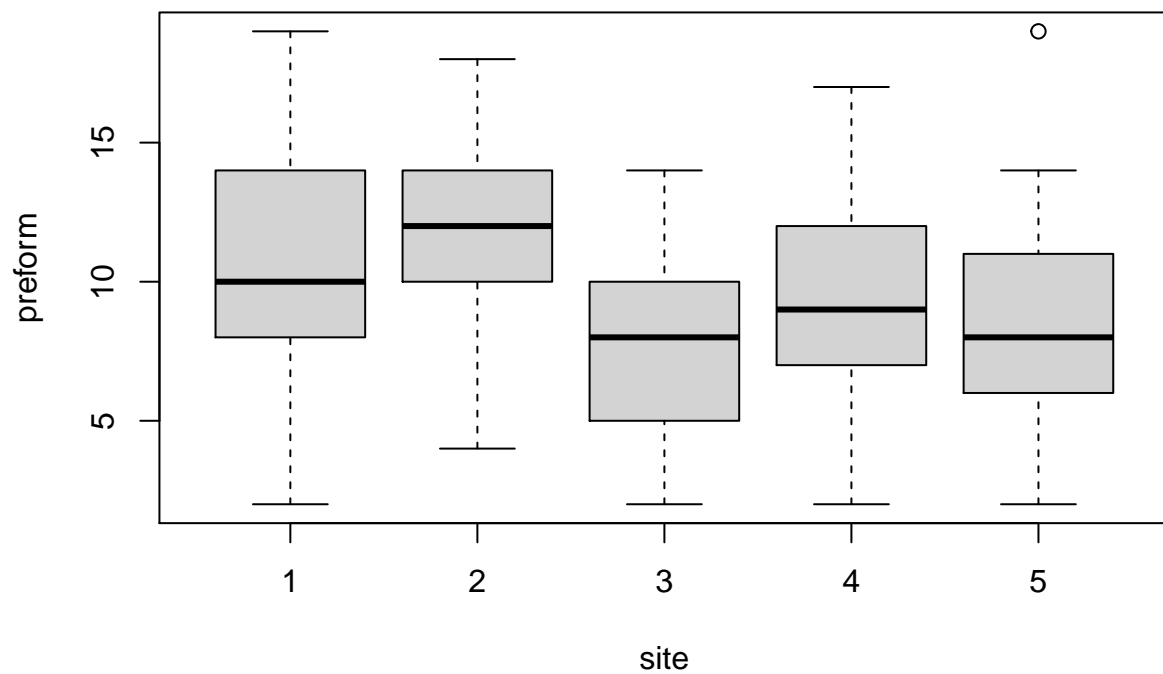
```
# trying to do more EDA to see if anything explains why the data is not linearly separable  
boxplot(prebody~site, data=sesame)
```



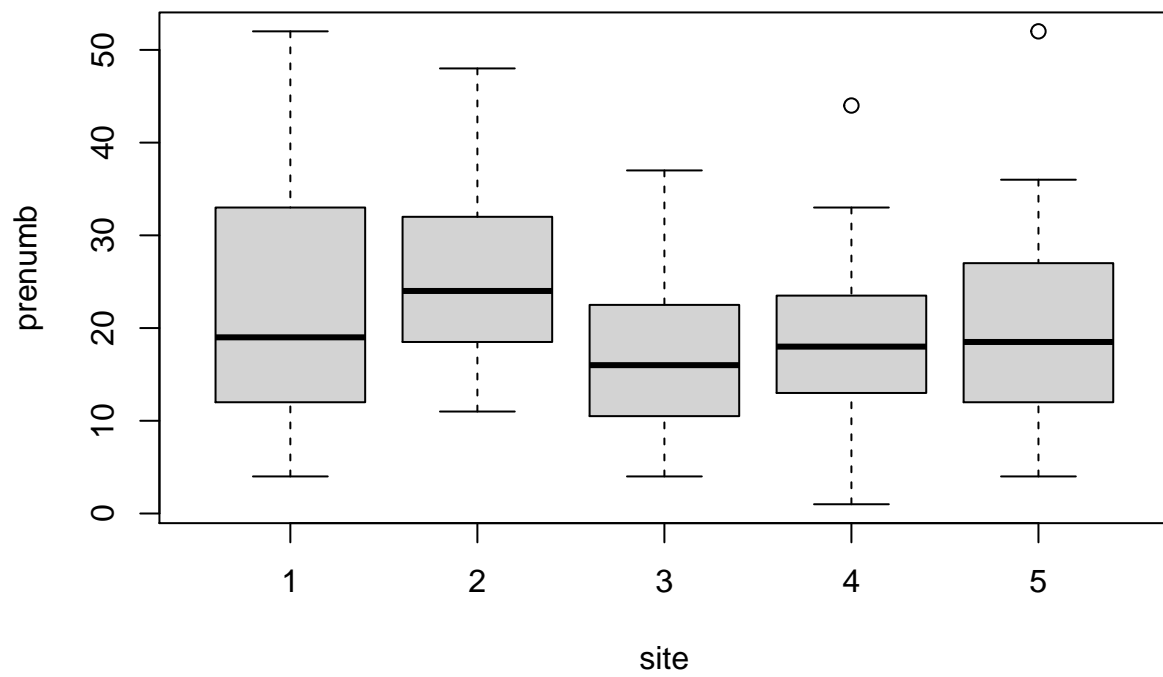
```
boxplot(prelet~site, data=sesame)
```



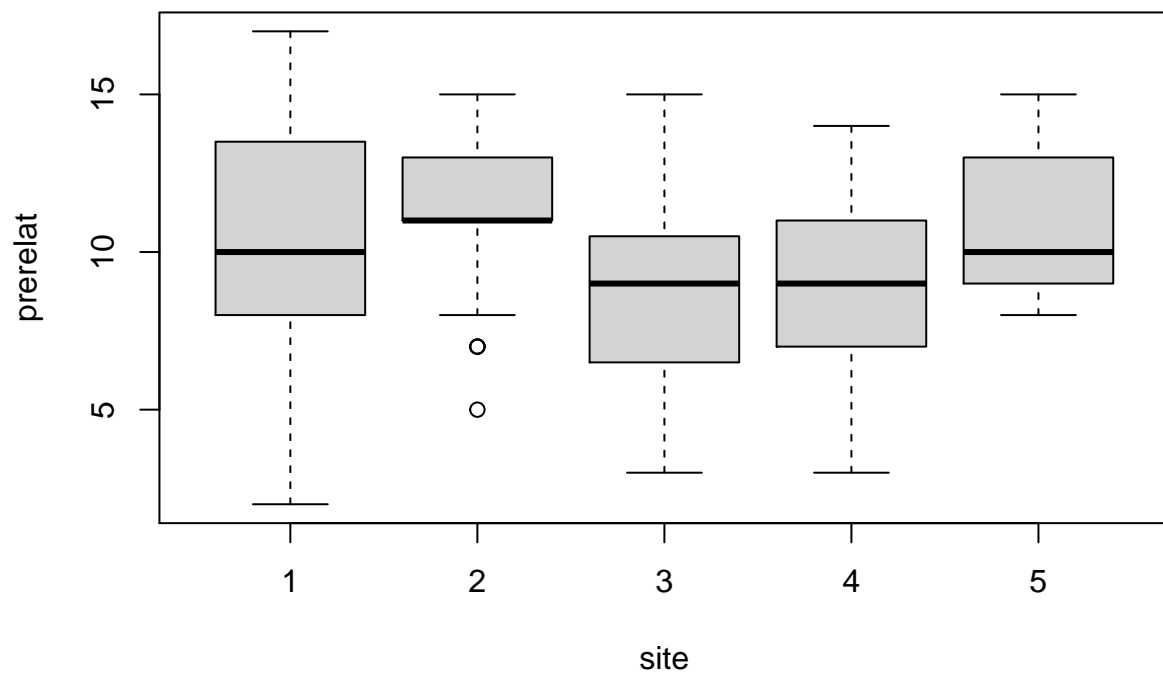
```
boxplot(preform~site, data=sesame)
```

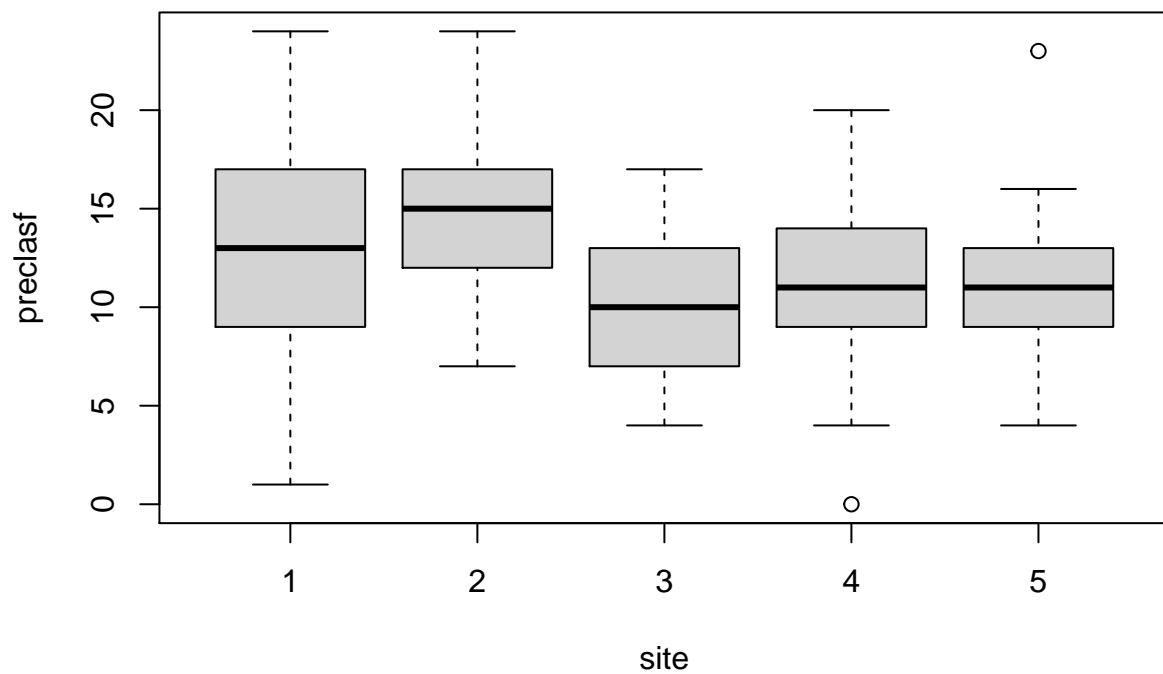
```
boxplot(preform~site, data=sesame)
```



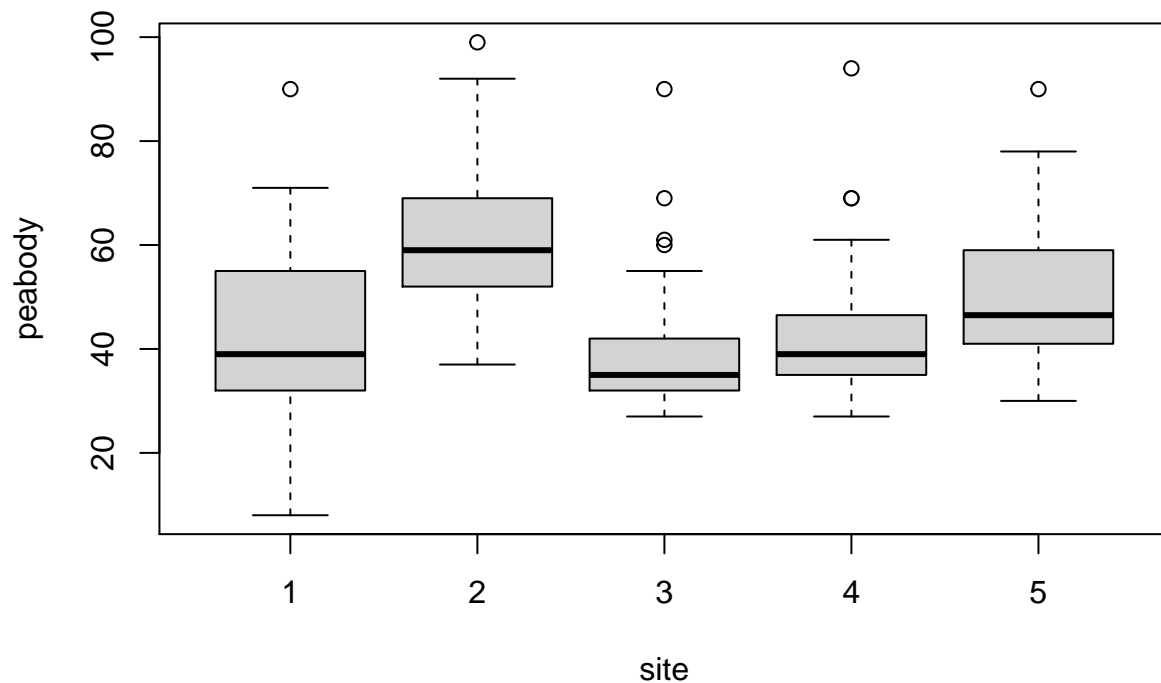
```
boxplot(prerelat~site, data=sesame)
```



```
boxplot(preclasf~site, data=sesame)
```



```
boxplot(peabody~site, data=sesame)
```



Trees

```
set.seed(3215)
#tree.data <- sesame %>%
# select(site, sex, age, viewcat, setting, viewenc, prebody, prelet, preform,
#        prenumb, prerelat, preclasf)

n <- nrow(sesame)
train.index <- sample(1:n, size = floor(0.7*n), replace=FALSE)
#train.tree <- tree.data[train.index,]
#test.tree <- tree.data[-train.index,]
# "viewcat", "setting", "viewenc",

# ,"prebody", "prelet","preform", "prenumb", "prerelat", "preclasf", "postbody", "postlet", "postform",

tree.features <- c("site", "age", "viewcat", "setting", "viewenc")

tree.data <- sesame[, tree.features]
train.data <- tree.data[train.index,]
test.data <- tree.data[-train.index,]

rf.tree<- randomForest(site~., data=tree.data, subset=train.index,
                        mtry=4, importance=TRUE)
```

```
importance(rf.tree)
```

```
##           1           2           3           4           5 MeanDecreaseAccuracy
## age      16.071180 11.5639655  1.533526 10.24956 -12.646482      15.477245
## viewcat   1.412544  8.8509648  4.714890 26.85440  -5.674656      19.084980
## setting   9.950184  0.7116516 11.570042 20.04012   6.879913      22.690481
## viewenc   1.061428 -1.4942151  6.283556 -2.36010  -3.154027       1.353554
##      MeanDecreaseGini
## age              67.80480
## viewcat          24.57478
## setting          10.77912
## viewenc          10.03605
```

```
rf.pred <- predict(rf.tree, newdata=test.data)
```

```
tree.conMatrix <- table(true=test.data[, "site"],
                        pred=rf.pred)
confusionMatrix(tree.conMatrix)
```

```
## Confusion Matrix and Statistics
```

```
##
##      pred
## true  1  2  3  4  5
##   1 10  2  3  3  0
##   2  4 10  1  1  2
##   3  2  3 11  3  0
##   4  1  1  3  5  2
##   5  0  1  3  0  1
```

```
## Overall Statistics
```

```
##
##              Accuracy : 0.5139
##              95% CI : (0.3931, 0.6335)
##      No Information Rate : 0.2917
##      P-Value [Acc > NIR] : 6.203e-05
```

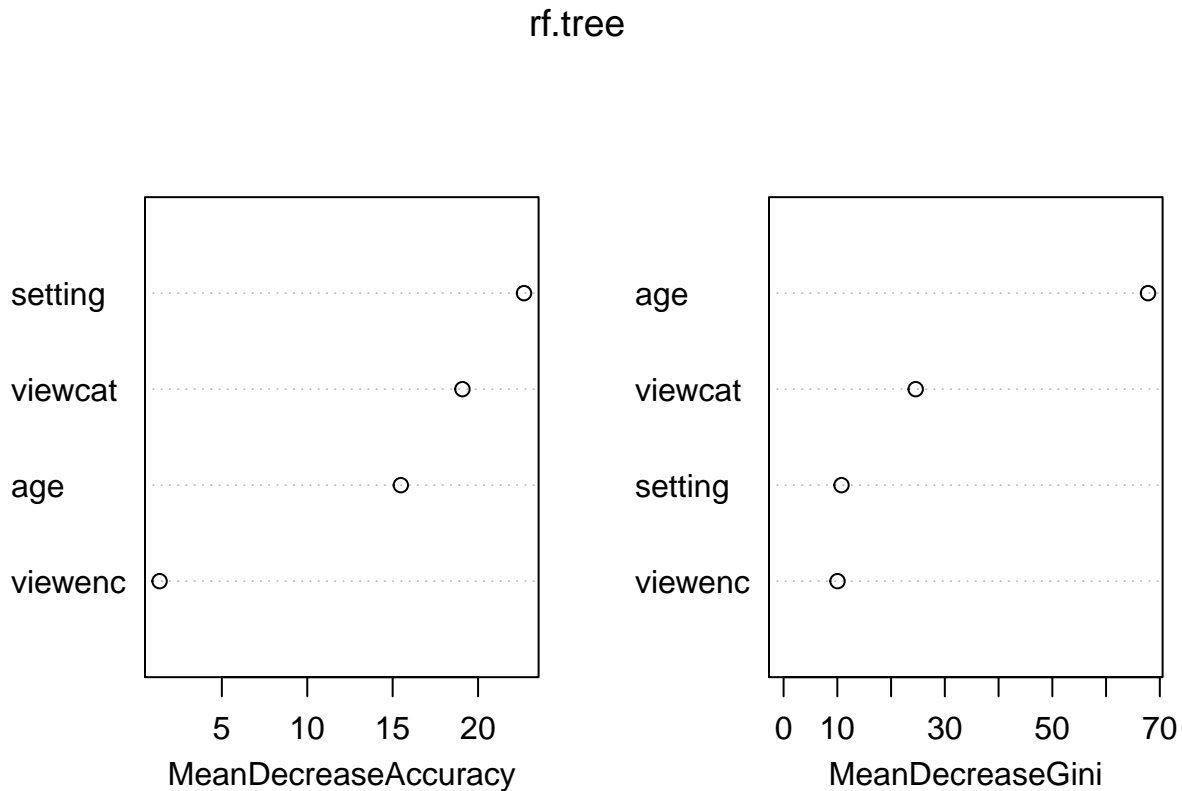
```
##
##              Kappa : 0.3706
```

```
## McNemar's Test P-Value : NA
```

```
## Statistics by Class:
```

```
##
##           Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.5882   0.5882   0.5238  0.41667  0.20000
## Specificity      0.8545   0.8545   0.8431  0.88333  0.94030
## Pos Pred Value   0.5556   0.5556   0.5789  0.41667  0.20000
## Neg Pred Value   0.8704   0.8704   0.8113  0.88333  0.94030
## Prevalence       0.2361   0.2361   0.2917  0.16667  0.06944
## Detection Rate   0.1389   0.1389   0.1528  0.06944  0.01389
## Detection Prevalence 0.2500  0.2500  0.2639  0.16667  0.06944
## Balanced Accuracy 0.7214   0.7214   0.6835  0.65000  0.57015
```

```
varImpPlot(rf.tree)
```



0.42 – 0.5139 (but not including the test scores.) around 0.45-0.48, when including the pretest scores.

As seen in the table above, there is a notable discrepancy in the number of observations that lay in classes 4 and 5 for the variable `site`. More specifically, in the training data, there are just 25 observations with a value of 4 for the variable `site` and just 13 observations with a value of 5 for the variable `site`. In other words, there are less disadvantaged rural children and disadvantaged Spanish speaking children.

Consequently, when we initially ran our random forest model, our model was performing worse for test observations that take on the values 4 or 5 for the variable `site`.

To remedy this problem, we decided to use Synthetic Minority Oversampling Technique (SMOTE). SMOTE works by generating new samples in the classes of the response variable that are less represented. These new samples are generated using linear combinations of the “k” nearest neighbors in a given class. In this instance, we set $k = 5$.

```
train.data$age <- as.numeric(train.data$age)
train.data$viewcat <- as.numeric(train.data$viewcat)
train.data$setting <- as.numeric(train.data$setting)
train.data$viewenc <- as.factor(train.data$viewenc)

test.data$age <- as.numeric(test.data$age)
test.data$viewcat <- as.numeric(test.data$viewcat)
test.data$setting <- as.numeric(test.data$setting)
test.data$viewenc <- as.factor(test.data$viewenc)
```

```
balanced.train.data <- SmoteClassif(site ~ ., train.data, k = 5, repl = FALSE, dist = "HEOM")
# k --> represents the number of nearest neighbors (5) used to generate new examples of the minority
# repl = FALSE --> cannot have repetition of examples when performing under-sampling by selecting amo
```

```
balanced.train.data %>%
  count(site)
```

```
##   site  n
## 1    1 34
## 2    2 34
## 3    3 34
## 4    4 33
## 5    5 34
```

```
rf.tree<- randomForest(site~., data=balanced.train.data,
                        mtry=4, importance=TRUE)
```

```
importance(rf.tree)
```

```
##           1           2           3           4           5 MeanDecreaseAccuracy
## age      15.112165 11.5044815 2.831522 12.2530454 12.35896      24.534350
## viewcat   2.482319 12.5671060 5.114628 28.5410065 19.94849      32.088862
## setting  14.672536  0.4233347 5.467389 22.8438707 32.97928      34.377861
## viewenc   5.604315 -2.5081007 6.322412  0.1078161 12.91417      9.960821
##           MeanDecreaseGini
## age              73.124641
## viewcat          29.595993
## setting           9.359814
## viewenc           9.254742
```

```
rf.pred <- predict(rf.tree, newdata=test.data)
```

```
tree.conMatrix <- table(true=test.data[, "site"],
                        pred=rf.pred)
confusionMatrix(tree.conMatrix)
```

```
## Confusion Matrix and Statistics
```

```
##
##      pred
## true 1 2 3 4 5
##    1 7 4 3 2 2
##    2 4 8 1 2 3
##    3 3 4 9 2 1
##    4 0 2 3 5 2
##    5 0 2 2 0 1
##
```

```
## Overall Statistics
```

```
##
##           Accuracy : 0.4167
##           95% CI : (0.3015, 0.5389)
##    No Information Rate : 0.2778
##    P-Value [Acc > NIR] : 0.007741
```



```
##
##          Kappa : 0.2539
##
## Mcnemar's Test P-Value : 0.576888
##
## Statistics by Class:
##
##          Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.50000   0.4000   0.5000  0.45455  0.11111
## Specificity      0.81034   0.8077   0.8148  0.88525  0.93651
## Pos Pred Value   0.38889   0.4444   0.4737  0.41667  0.20000
## Neg Pred Value   0.87037   0.7778   0.8302  0.90000  0.88060
## Prevalence       0.19444   0.2778   0.2500  0.15278  0.12500
## Detection Rate   0.09722   0.1111   0.1250  0.06944  0.01389
## Detection Prevalence 0.25000   0.2500   0.2639  0.16667  0.06944
## Balanced Accuracy 0.65517   0.6038   0.6574  0.66990  0.52381
```

While the `SmoteClassif()` function certainly did its job by balancing out the number of observations for each value of `site` in the training data, the new random forest model (fitted to this new data set) is less accurate and sees little improvement in the detection of `site` values of 4 and 5.

```
# set.seed(3215)
#
# # ,"prebody", "prelet","preform", "prenumb", "prerelat", "preclasf", "postbody", "postlet", "postform"
#
#
#
# features <- c("site", "age", "viewcat", "setting", "viewenc","prebody", "prelet","preform", "prenumb"
#
# tree.2 <- sesame[, features]
# train.2 <- tree.2[train.index,]
# test.2 <- tree.2[-train.index,]
#
# boost.tree <- gbm(site ~., data=train.2,
#                   distribution="multinomial", n.trees=5000,
#                   interaction.depth=1)
#
# #y.boost <- table(true=test.2[, "site"],
# #                 pred=predict(boost.tree, newdata=test.2))
#
# boost.conMatrix <- table(true=test.2$site,
#                           pred=predict(boost.tree, newdata=test.2))
# confusionMatrix(boost.conMatrix)
```

Questions for OH:

should we transform regular?

Both linear and radial kernels never output predictions for 4 & 5?

polynomial kernel? Which variables to give polynomial terms

use PCA to perform feature selection?

feature selections for SVM in general?

how to interpret the confusion matrix tables for SVM & Trees

How to interpret the importance variance for multiclass classification

interpretations about the dataset, using the bad performance of the classifiers