

Analyzing and Predicting the Relationships between Sesame Street Viewership and Test Scores among School Children

Angela Wang, QiHan Zhou, Matthew Murray, Michelle Mao, Sara Lemus, Sophie Dalldorf

12/9/2021

Introduction

Research Question #1

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)?

Methodology

Question 1

We decided to utilize three different types of models to gain inference and predict the difference in test scores that occurs after subjects watch Sesame Street: (1) least-squares regression model (2) ridge regression model and (3) regression tree model. There are six different test scores in the data set, so we fit three models for each difference in test score variable. Each model took the following variables as covariates: `site`, `sex`, `viewcat`, `setting`, `viewenc`. Before creating these models, we factored those variables to encode them as categoricals. Initially, our least-squares and ridge regression models also took into account all possible two-way interaction terms between the variables. However, we noticed that both the Akaike Information Criterion (AIC) and adjusted- R^2 values were higher for models that did not include any interaction effects. That reason, coupled with the lack of apparent interaction effects in our exploratory data analysis and the fact that we wanted our linear models to be very interpretable, is why we decided not to include any interaction effects in our final linear models.

One problem that we envisioned when evaluating 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.

We first decided to use least-squares regression models due to the fact that they provide apt inference into the relationship between the covariates and the response variable. Thereafter, we decided to use a ridge regression model due to the fact that ridge regression often provides performance improvements by shrinking slope coefficients. While shrinkage may introduce bias to a model, it decreases the variance and increases the precision of the slope coefficient estimates. The shrinkage is achieved by applying a shrinkage parameter, λ , to the Euclidean norm of a slope coefficient. Doing so slightly increases the bias of our model (as least-squares regression coefficient estimates are unbiased) but can also significantly decrease the variance (and increase

the precision) of our regression coefficients. This slight increase in bias but significant decrease in variance usually decreases the MSE of a model.

We tuned our λ parameter using 10-fold cross validation. More specifically, we computed the cross-validation error rate for our model for a grid of λ values. Thereafter, we selected the λ value for which the cross-validation error is the smallest.

Initially, we were inclined to use least absolute shrinkage and selection operator (LASSO) regression. The main advantage of LASSO regression over ridge regression is that LASSO regression performs variable selection by setting the slope coefficients of inert predictors to 0. The reason why we initially thought that LASSO regression would work better than ridge regression is that in most of our linear models, only a small subset of variables are significant at a 95% significance level. However, our ridge regression models performed marginally better than our LASSO regression models, so for this reason, we decided to report the MSE's for the ridge regression models.

~Talk about why we chose regression tree models~

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

Results

Linear Regression Models

Table 1: Test Metrics

Response	Least.Squares.Reggression.Test.MSE	Ridge.Reggression.Test.MSE	Regression.Tree.Test.MSE
bodyDiff	24.01	21.61	20.60
letDiff	14.31	14.20	15.41
formDiff	13.26	12.83	14.92
numbDiff	15.51	14.63	15.91
relatDiff	18.94	19.86	19.89
clasfDiff	48.40	44.26	45.53

From the results of the above table, one can see that the ridge regression models have the best performance, although the performance of the regression trees are very similar. For each response variable except `bodyDiff`, the ridge regression model reports the lowest test MSE, although the difference in performance between the ridge regression and the regression tree is very marginal.

Conclusion

Appendix

```
##
## Call:
## lm(formula = bodyDiff ~ (site + sex + age + viewcat + setting +
##      viewenc), data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3707  -2.7927   0.0525   2.5545  13.1411
```

```
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.42668   3.06383   5.361 2.92e-07 ***
## site2       0.58666   0.99737   0.588 0.55724
## site3       2.44827   0.96820   2.529 0.01244 *
## site4       2.96605   1.10488   2.685 0.00805 **
## site5       3.29854   1.45590   2.266 0.02485 *
## sex2        -1.00041   0.66732  -1.499 0.13586
## age         -0.08726   0.05558  -1.570 0.11845
## viewcat2    -0.60564   1.12908  -0.536 0.59244
## viewcat3     1.30568   1.09680   1.190 0.23568
## viewcat4     2.18215   1.12555   1.939 0.05434 .
## setting2    -1.02953   0.77713  -1.325 0.18718
## viewenc2    -0.37832   0.84111  -0.450 0.65349
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.253 on 156 degrees of freedom
## Multiple R-squared:  0.1585, Adjusted R-squared:  0.09917
## F-statistic: 2.671 on 11 and 156 DF,  p-value: 0.00362

## [1] 976.6991

## [1] 24.01045

## [1] 21.60675

##
## Call:
## lm(formula = letDiff ~ (site + sex + age + viewcat + setting +
##       viewenc), data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.2823  -2.9776  -0.3663   2.9275  11.1426
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.52353   3.32595   3.164 0.00187 **
## site2       3.08586   1.08270   2.850 0.00496 **
## site3      -2.85941   1.05103  -2.721 0.00726 **
## site4      -1.07861   1.19940  -0.899 0.36989
## site5      -0.38501   1.58046  -0.244 0.80786
## sex2        0.37911   0.72441   0.523 0.60148
## age         0.05050   0.06034   0.837 0.40390
## viewcat2     1.59616   1.22568   1.302 0.19474
## viewcat3     4.85135   1.19064   4.075 7.32e-05 ***
## viewcat4     4.77123   1.22184   3.905 0.00014 ***
## setting2     0.70346   0.84362   0.834 0.40563
## viewenc2    -1.56705   0.91307  -1.716 0.08810 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4.617 on 156 degrees of freedom
## Multiple R-squared: 0.37, Adjusted R-squared: 0.3256
## F-statistic: 8.33 on 11 and 156 DF, p-value: 2.043e-11

## [1] 1004.281

## [1] 14.31277

## [1] 14.19572

##
## Call:
## lm(formula = formDiff ~ (site + sex + age + viewcat + setting +
##      viewenc), data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.3637  -2.5344   0.2658   2.7054  14.1942
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.04193    3.16388   5.070 1.11e-06 ***
## site2        0.97029    1.02994   0.942  0.3476
## site3        0.73495    0.99982   0.735  0.4634
## site4        0.92962    1.14096   0.815  0.4164
## site5        1.00499    1.50344   0.668  0.5048
## sex2         0.09439    0.68911   0.137  0.8912
## age        -0.05234    0.05740  -0.912  0.3632
## viewcat2     0.99101    1.16595   0.850  0.3966
## viewcat3     1.65614    1.13262   1.462  0.1457
## viewcat4     2.47768    1.16230   2.132  0.0346 *
## setting2    -0.08994    0.80251  -0.112  0.9109
## viewenc2    -0.42935    0.86858  -0.494  0.6218
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.392 on 156 degrees of freedom
## Multiple R-squared: 0.06302, Adjusted R-squared: -0.003048
## F-statistic: 0.9539 on 11 and 156 DF, p-value: 0.4909

## [1] 987.4956

## [1] 13.25639

## [1] 12.82502

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

```

## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.6433  -2.6010   0.1375   2.3541  10.3247
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 15.388591   3.084422   4.989 1.6e-06 ***
## site2       2.561427   1.004073   2.551 0.01170 *
## site3       0.768616   0.974707   0.789 0.43157
## site4       1.151651   1.112304   1.035 0.30210
## site5       1.946294   1.465685   1.328 0.18615
## sex2        0.013992   0.671801   0.021 0.98341
## age         0.009505   0.055956   0.170 0.86534
## viewcat2    1.841715   1.136668   1.620 0.10719
## viewcat3    3.199614   1.104174   2.898 0.00430 **
## viewcat4    3.636674   1.133111   3.209 0.00161 **
## setting2    0.596707   0.782354   0.763 0.44679
## viewenc2    0.589407   0.846765   0.696 0.48742
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.281 on 156 degrees of freedom
## Multiple R-squared:  0.139, Adjusted R-squared:  0.07825
## F-statistic: 2.289 on 11 and 156 DF, p-value: 0.01273

## [1] 978.9495

## [1] 15.5095

## [1] 14.62731

##
## Call:
## lm(formula = relatDiff ~ (site + sex + age + viewcat + setting +
##      viewenc), data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.9246  -2.7114   0.3439   2.2185  14.4415
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.48969   3.17203   5.829 3.1e-08 ***
## site2      -0.05900   1.03259  -0.057  0.9545
## site3       1.66242   1.00239   1.658  0.0992 .
## site4       2.41955   1.14390   2.115  0.0360 *
## site5      -0.40561   1.50731  -0.269  0.7882
## sex2        0.31932   0.69088   0.462  0.6446
## age        -0.09729   0.05755  -1.691  0.0929 .
## viewcat2     1.05190   1.16895   0.900  0.3696
## viewcat3     0.96616   1.13554   0.851  0.3962
## viewcat4     2.95954   1.16529   2.540  0.0121 *
## setting2    -0.60666   0.80457  -0.754  0.4520

```

```

## viewenc2      0.19462      0.87082      0.223      0.8234
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.403 on 156 degrees of freedom
## Multiple R-squared:  0.1072, Adjusted R-squared:  0.04421
## F-statistic: 1.702 on 11 and 156 DF,  p-value: 0.07739

## [1] 988.3598

## [1] 18.94131

## [1] 19.86283

##
## Call:
## lm(formula = clasfDiff ~ (site + sex + age + viewcat + setting +
##      viewenc), data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.1077  -4.0442   0.2665   3.8807  14.5260
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.951998   4.455915   2.458  0.0151 *
## site2        1.799749   1.450536   1.241  0.2166
## site3        0.420798   1.408112   0.299  0.7655
## site4        2.040728   1.606891   1.270  0.2060
## site5        1.497017   2.117404   0.707  0.4806
## sex2         1.500233   0.970519   1.546  0.1242
## age         -0.008915   0.080838  -0.110  0.9123
## viewcat2     3.041609   1.642090   1.852  0.0659 .
## viewcat3     3.072479   1.595147   1.926  0.0559 .
## viewcat4     4.060404   1.636951   2.480  0.0142 *
## setting2     0.477618   1.130229   0.423  0.6732
## viewenc2    -1.088979   1.223281  -0.890  0.3747
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.185 on 156 degrees of freedom
## Multiple R-squared:  0.09426, Adjusted R-squared:  0.03039
## F-statistic: 1.476 on 11 and 156 DF,  p-value: 0.1455

## [1] 1102.553

## [1] 48.398

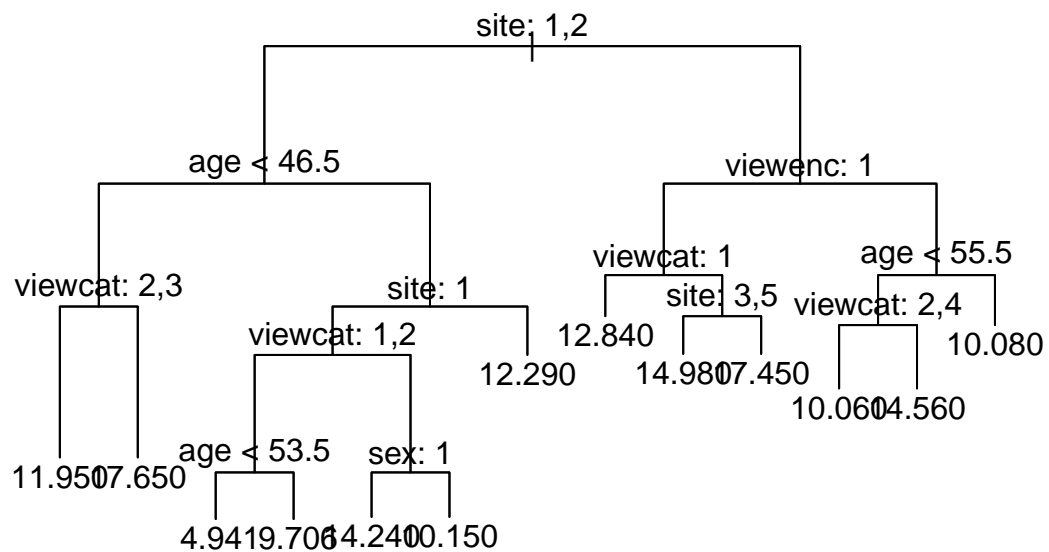
## [1] 44.26185

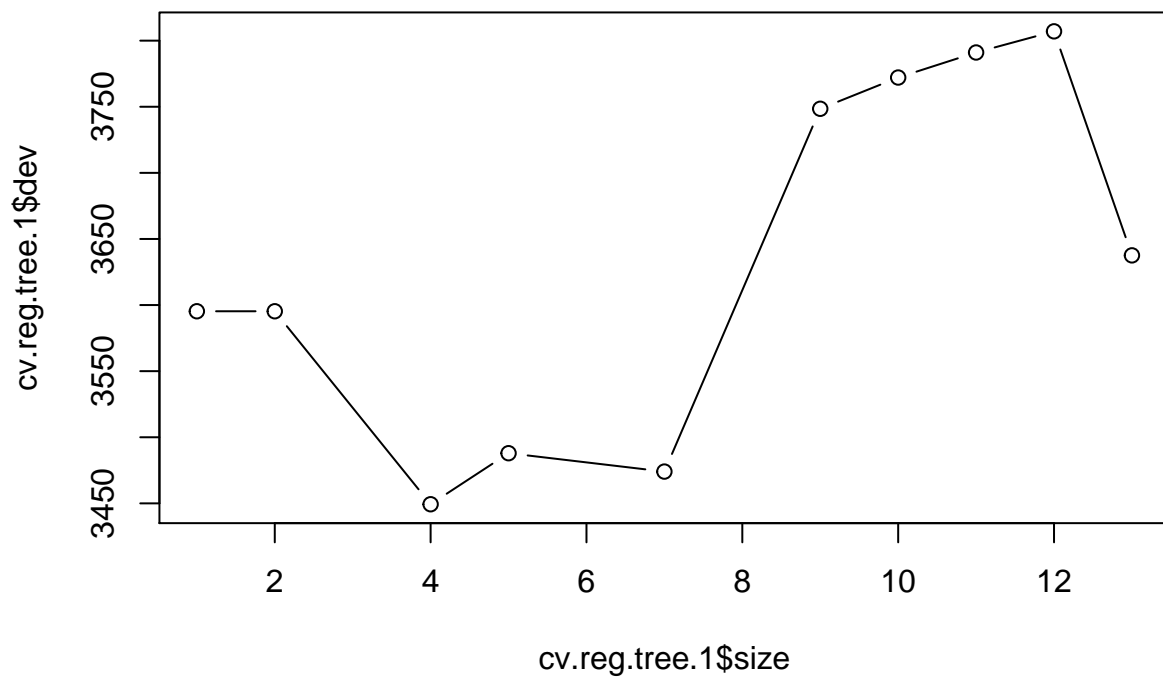
```

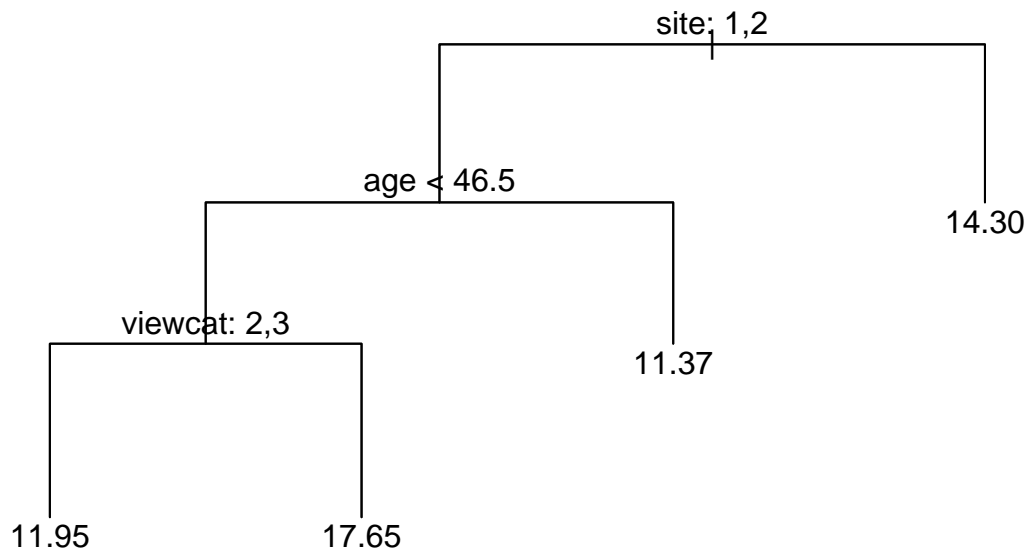
Regression Tree Models

Model 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
```





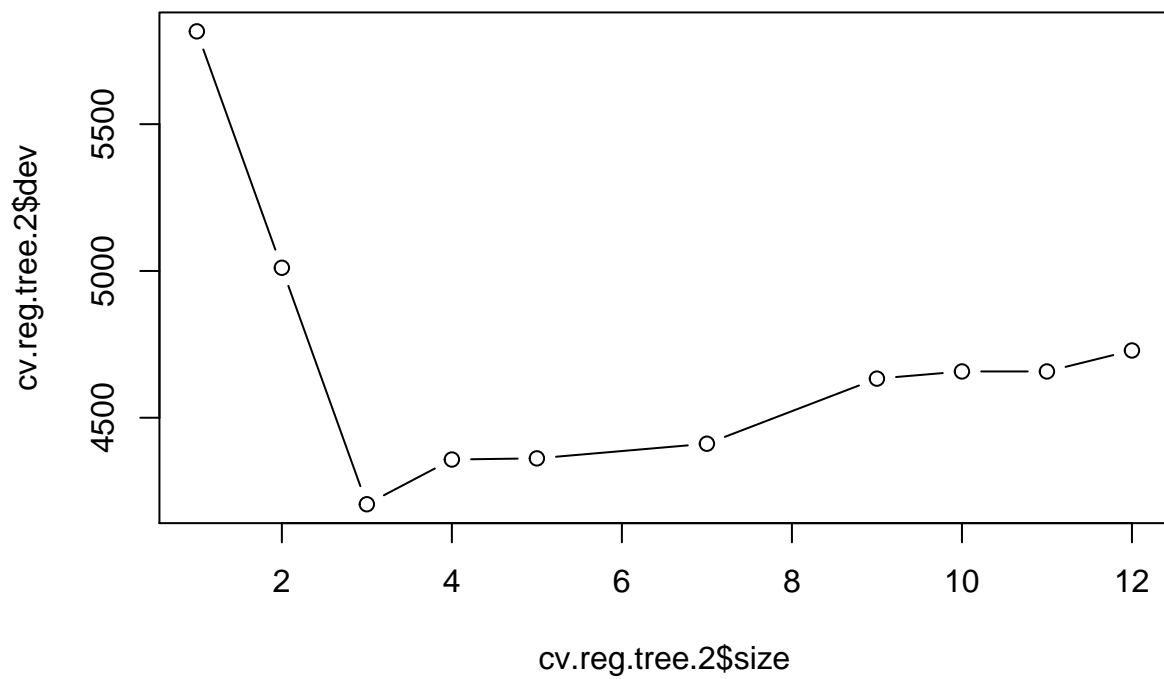


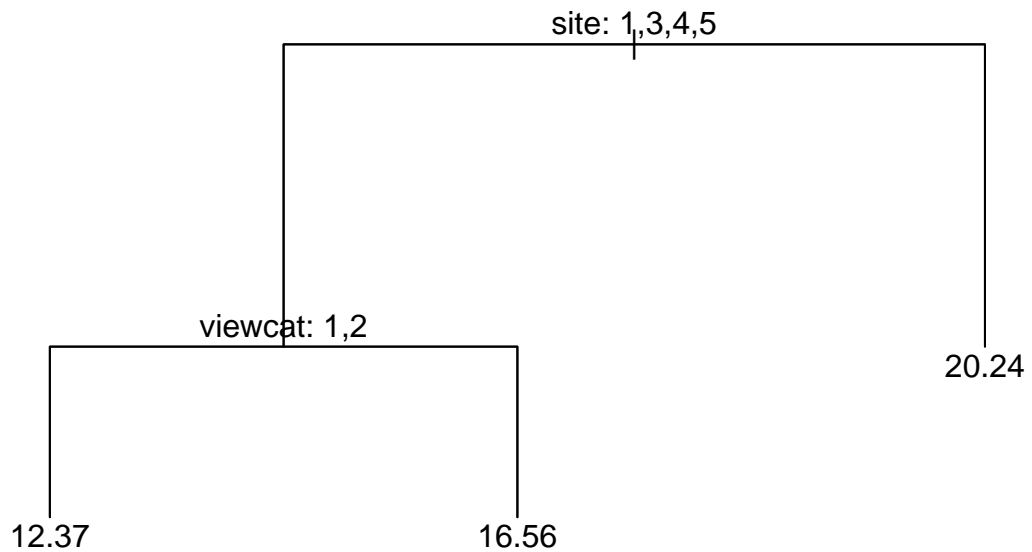
```
## [1] 20.60159
```

Model 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
```



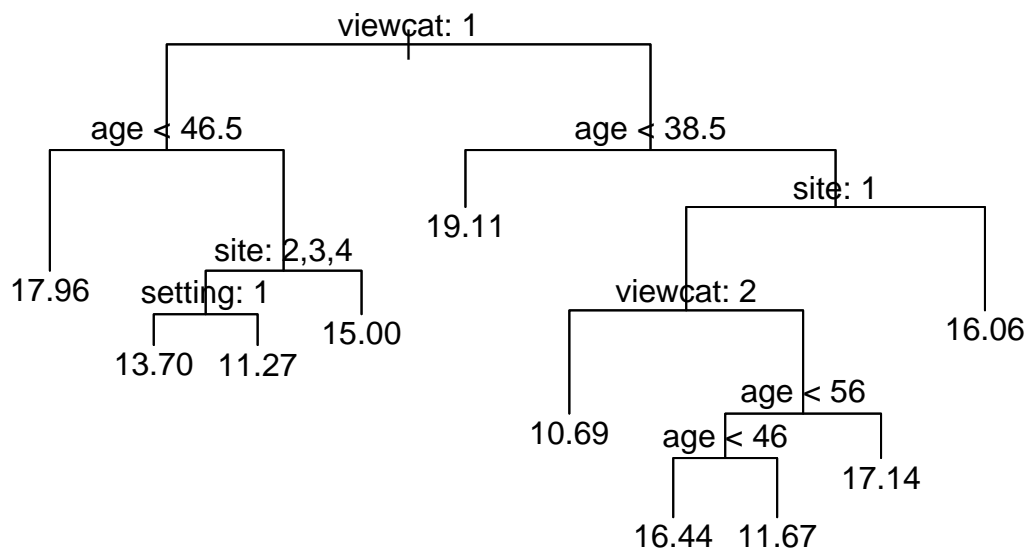


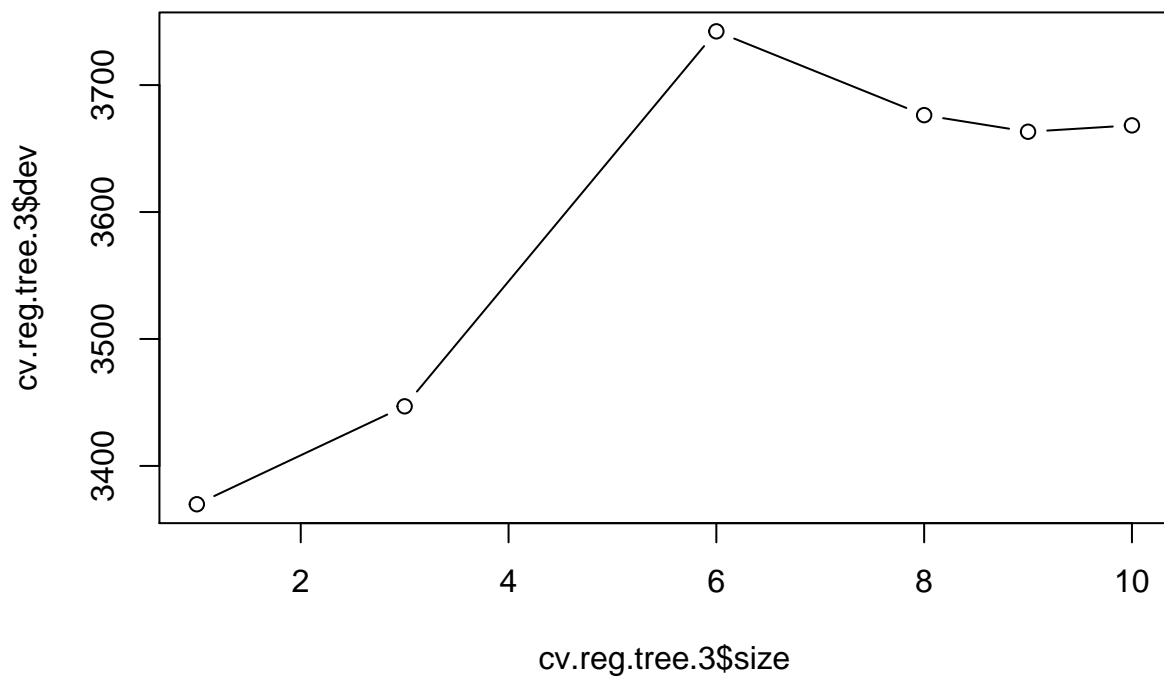


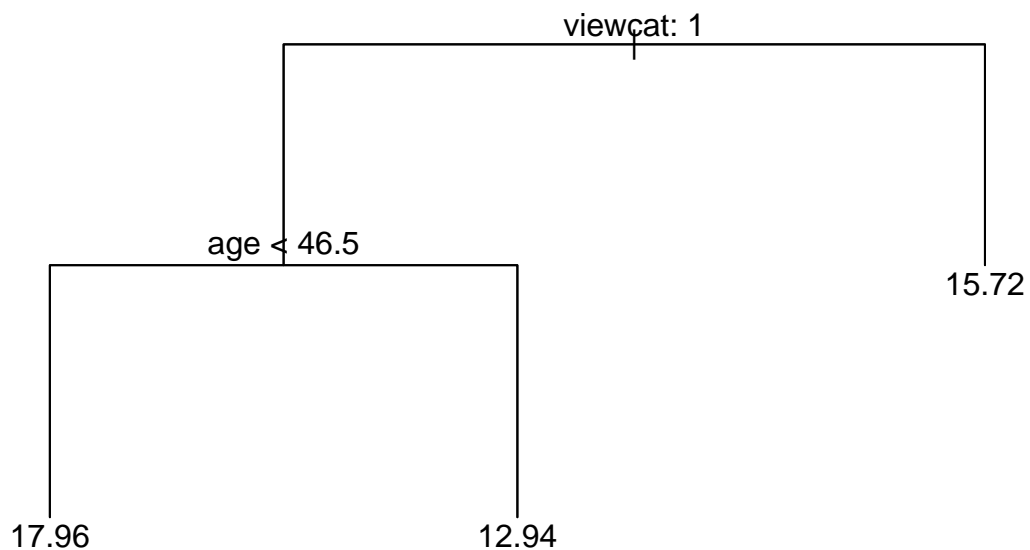
```
## [1] 15.4124
```

Model 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
```



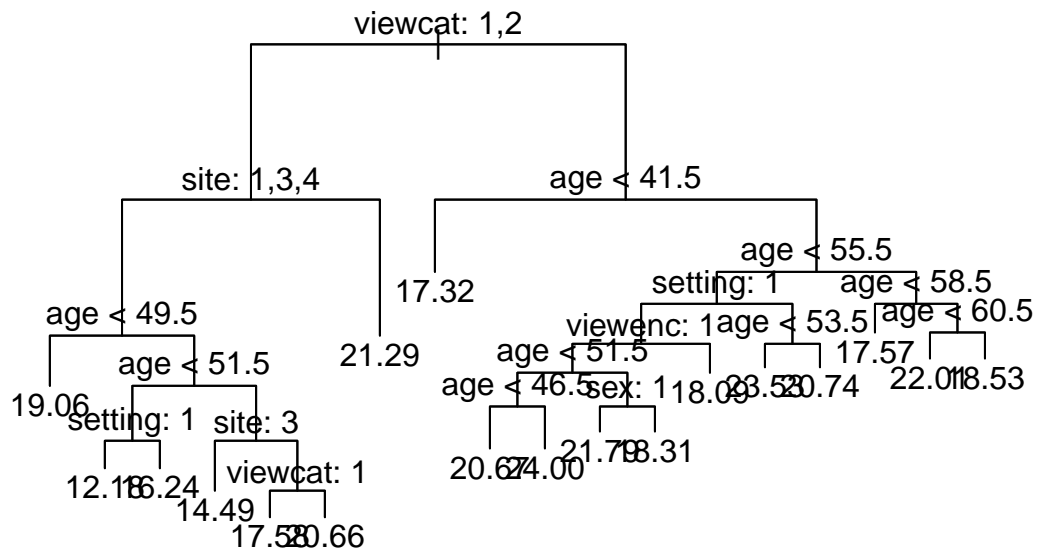


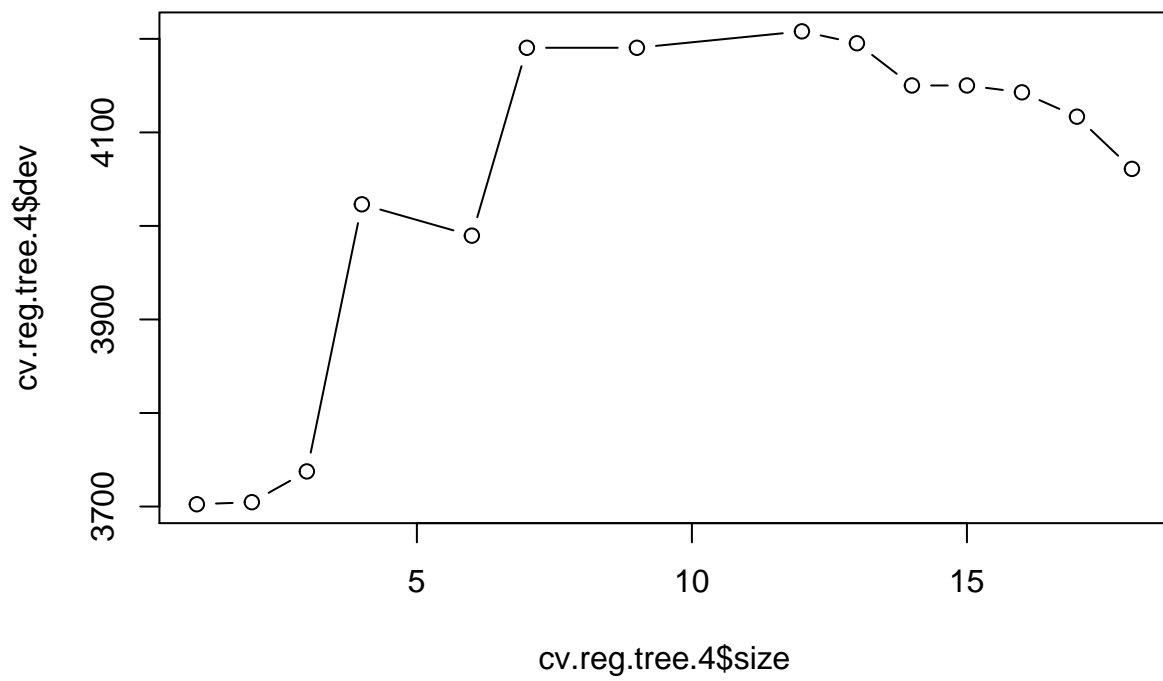


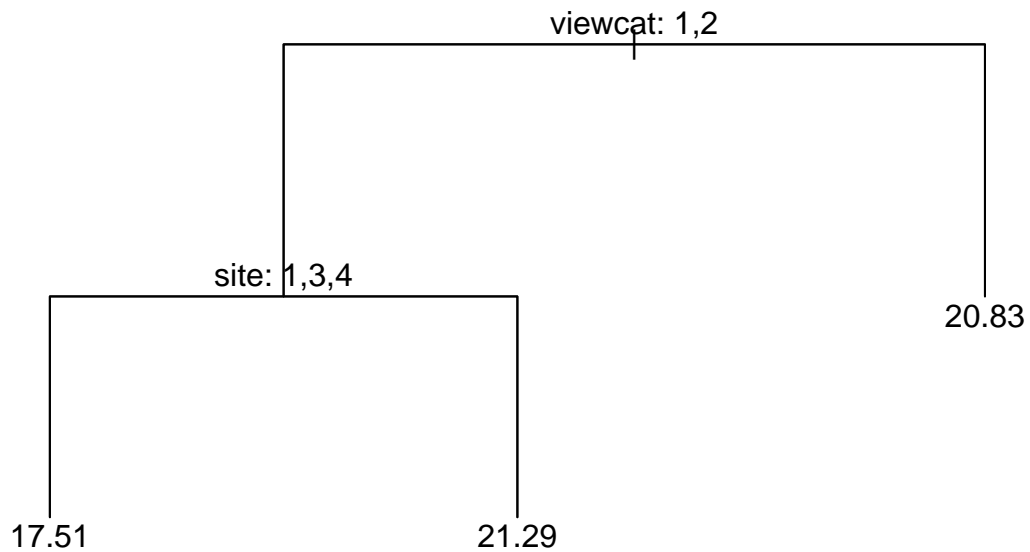
```
## [1] 14.92273
```

Model 4

```
##
## Regression tree:
## tree(formula = numbDiff ~ 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
```



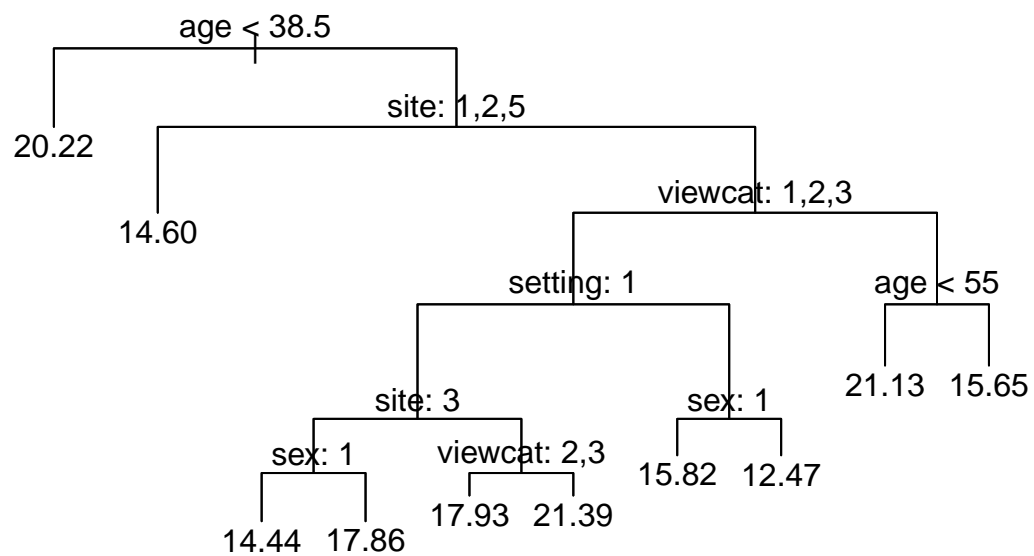


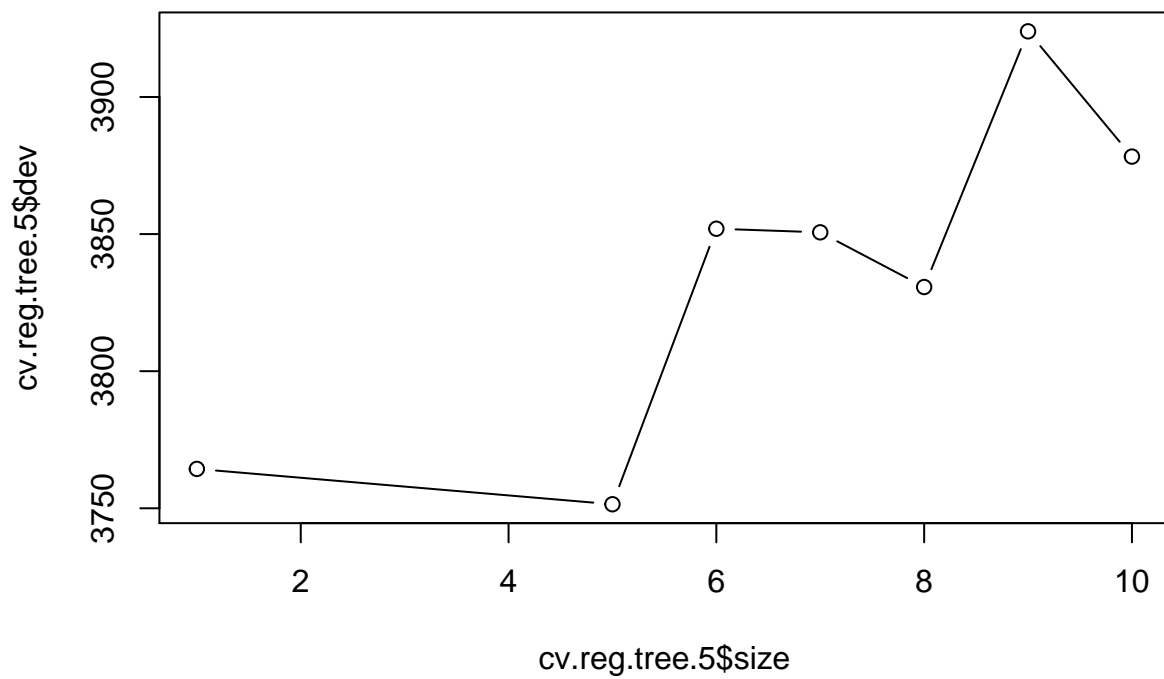


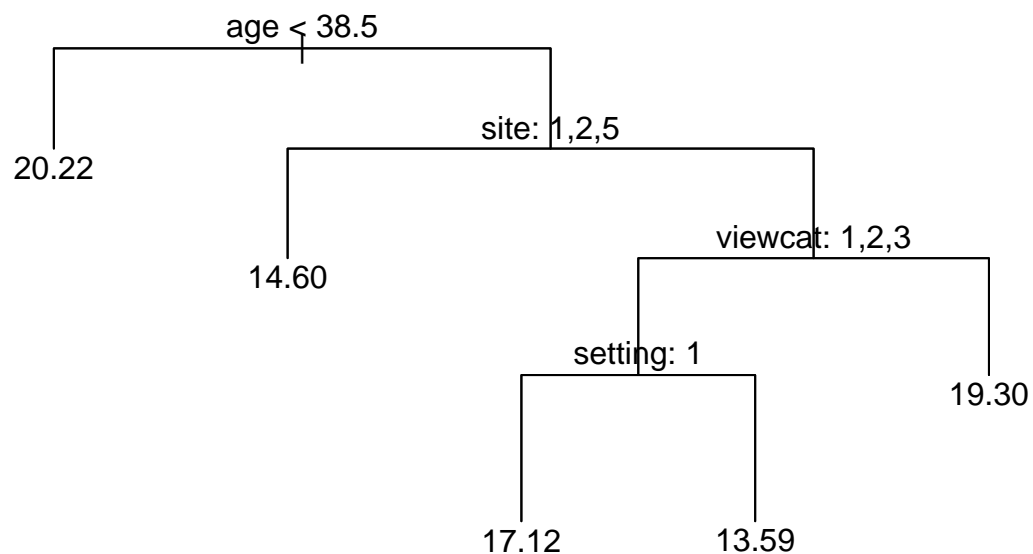
```
## [1] 15.90771
```

Model 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
```



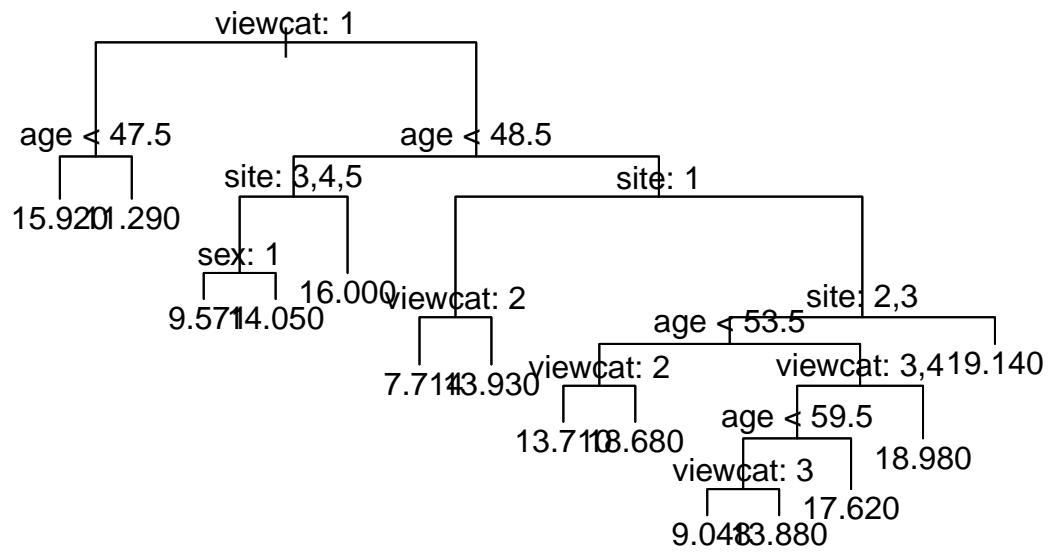


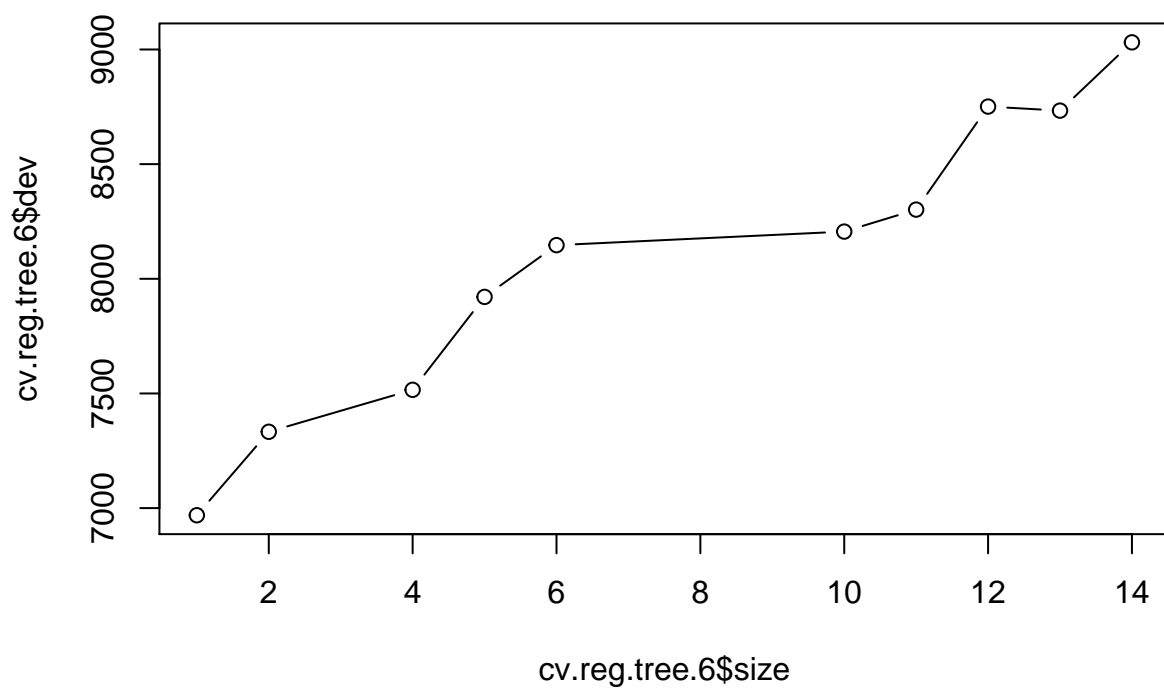


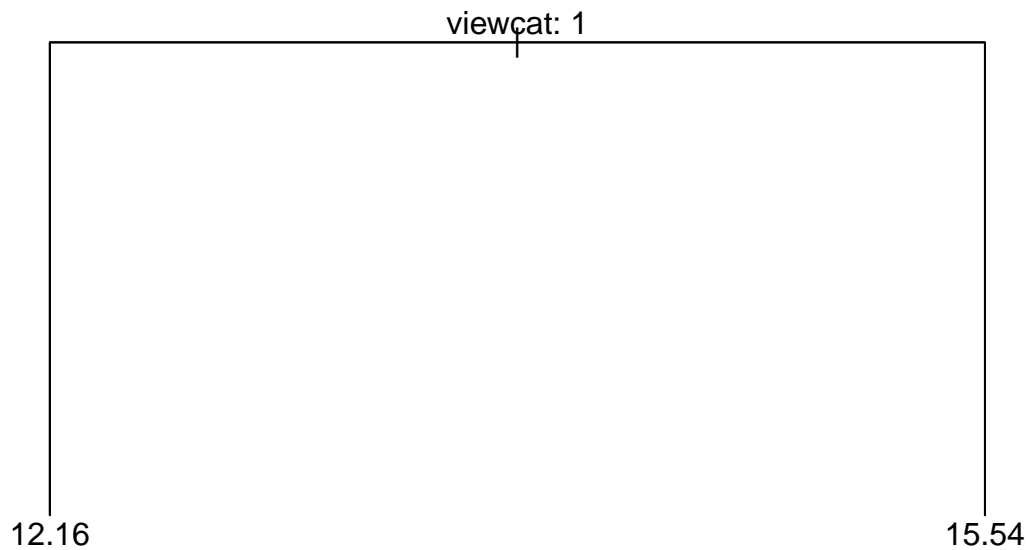
```
## [1] 19.88506
```

Model 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
```







```
## [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?

SVM

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

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

```
##
##      pred
## true  1  2  3  4  5
##    1  2  5 13  0  0
##    2  0  8  5  0  0
##    3  1  1 14  0  0
##    4  0  4 14  0  0
##    5  0  1  4  0  0
##
```



```

## Overall Statistics
##
##           Accuracy : 0.3333
##           95% CI : (0.2266, 0.4543)
##       No Information Rate : 0.6944
##       P-Value [Acc > NIR] : 1
##
##           Kappa : 0.1523
##
##  McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.66667  0.4211  0.2800      NA      NA
## Specificity      0.73913  0.9057  0.9091    0.75  0.93056
## Pos Pred Value   0.10000  0.6154  0.8750      NA      NA
## Neg Pred Value   0.98077  0.8136  0.3571      NA      NA
## Prevalence       0.04167  0.2639  0.6944    0.00  0.00000
## Detection Rate   0.02778  0.1111  0.1944    0.00  0.00000
## Detection Prevalence 0.27778  0.1806  0.2222    0.25  0.06944
## Balanced Accuracy 0.70290  0.6634  0.5945      NA      NA

## Confusion Matrix and Statistics
##
##      pred
## true  1  2  3  4  5
##   1   7  3 10  0  0
##   2   3  6  4  0  0
##   3   0  2 14  0  0
##   4   2  4 12  0  0
##   5   0  1  4  0  0
##
## Overall Statistics
##
##           Accuracy : 0.375
##           95% CI : (0.2636, 0.497)
##       No Information Rate : 0.6111
##       P-Value [Acc > NIR] : 1
##
##           Kappa : 0.1964
##
##  McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.58333  0.37500  0.3182      NA      NA
## Specificity      0.78333  0.87500  0.9286    0.75  0.93056
## Pos Pred Value   0.35000  0.46154  0.8750      NA      NA
## Neg Pred Value   0.90385  0.83051  0.4643      NA      NA
## Prevalence       0.16667  0.22222  0.6111    0.00  0.00000
## Detection Rate   0.09722  0.08333  0.1944    0.00  0.00000
## Detection Prevalence 0.27778  0.18056  0.2222    0.25  0.06944

```

```
## Balanced Accuracy      0.68333  0.62500  0.6234      NA      NA
```

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

```
##
##      pred
## true  1  2  3  4  5
##      1  1  7 12  0  0
##      2  1  8  4  0  0
##      3  3  1 12  0  0
##      4  3  4 11  0  0
##      5  0  1  4  0  0
##
```

``` ## Overall Statistics ```

```
##
##              Accuracy : 0.2917
##              95% CI : (0.1905, 0.4107)
##      No Information Rate : 0.5972
##      P-Value [Acc > NIR] : 1
##
##              Kappa : 0.0962
##
##      McNemar's Test P-Value : NA
##
```

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

```
##
##              Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.12500   0.3810   0.2791      NA      NA
## Specificity      0.70312   0.9020   0.8621   0.75  0.93056
## Pos Pred Value   0.05000   0.6154   0.7500      NA      NA
## Neg Pred Value   0.86538   0.7797   0.4464      NA      NA
## Prevalence       0.11111   0.2917   0.5972   0.00  0.00000
## Detection Rate   0.01389   0.1111   0.1667   0.00  0.00000
## Detection Prevalence 0.27778   0.1806   0.2222   0.25  0.06944
## Balanced Accuracy 0.41406   0.6415   0.5706      NA      NA
```

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

```
##
##      pred
## true  1  2  3  4  5
##      1  9  2  9  0  0
##      2  4  5  4  0  0
##      3  3  1 12  0  0
##      4  3  3 12  0  0
##      5  0  1  4  0  0
##
```

``` ## Overall Statistics ```

```
##
##              Accuracy : 0.3611
##              95% CI : (0.2512, 0.4829)
##      No Information Rate : 0.5694
##      P-Value [Acc > NIR] : 0.9999
##
##              Kappa : 0.1703
##
```

```
## McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##          Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity      0.4737  0.41667  0.2927      NA      NA
## Specificity      0.7925  0.86667  0.8710      0.75  0.93056
## Pos Pred Value   0.4500  0.38462  0.7500      NA      NA
## Neg Pred Value   0.8077  0.88136  0.4821      NA      NA
## Prevalence       0.2639  0.16667  0.5694      0.00  0.00000
## Detection Rate   0.1250  0.06944  0.1667      0.00  0.00000
## Detection Prevalence 0.2778  0.18056  0.2222      0.25  0.06944
## Balanced Accuracy 0.6331  0.64167  0.5818      NA      NA
```

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
```

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

```



```

## 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
## Pos Pred Value   0.05000  0.23077  0.25000  0.6111  0.00000
## Neg Pred Value   0.96154  0.98305  0.64286  0.5000  0.95522

```

```

## Prevalence          0.04167  0.05556  0.33333  0.5278  0.04167
## Detection Rate      0.01389  0.04167  0.05556  0.1528  0.00000
## Detection Prevalence 0.27778  0.18056  0.22222  0.2500  0.06944
## Balanced Accuracy    0.52899  0.80147  0.45833  0.5418  0.46377

## Confusion Matrix and Statistics
##
##      pred
## true  1  2  3  4  5
##      1  6  2  5  1  6
##      2  4  2  3  0  4
##      3  1  0 11  1  3
##      4  3  2  8  1  4
##      5  1  1  2  0  1
##
## Overall Statistics
##
##              Accuracy : 0.2917
##              95% CI : (0.1905, 0.4107)
##      No Information Rate : 0.4028
##      P-Value [Acc > NIR] : 0.981095
##
##              Kappa : 0.1226
##
## Mcnemar's Test P-Value : 0.006726
##
## Statistics by Class:
##
##              Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
## Sensitivity          0.40000  0.28571  0.3793  0.33333  0.05556
## Specificity          0.75439  0.83077  0.8837  0.75362  0.92593
## Pos Pred Value       0.30000  0.15385  0.6875  0.05556  0.20000
## Neg Pred Value       0.82692  0.91525  0.6786  0.96296  0.74627
## Prevalence           0.20833  0.09722  0.4028  0.04167  0.25000
## Detection Rate       0.08333  0.02778  0.1528  0.01389  0.01389
## Detection Prevalence 0.27778  0.18056  0.2222  0.25000  0.06944
## Balanced Accuracy    0.57719  0.55824  0.6315  0.54348  0.49074

```

In order to address our second research question, predicting whether a child came from an disadvantaged background or not based on their pretest scores and demographic information, we utilized a support vector machine (SVM). Our model uses the female, male, age, and all pretest score variables to predict our response variable, site. Since our response variable is a categorical variable, a SVM is a valid choice to answer our research question. We also implemented a classification tree to answer this question as well; however, this model performed poorly on our data. Thus, a SVM was the most appropriate model choice for our research goals. Our full model formula is: (add formula)

We split our data into a 70% training set and 30% testing set and analyzed the performance of our model on the test set. To improve our model's predictive power, we implemented a variety of different methods. We tested our model using linear, radial, and sigmoid kernels and compared the predictive accuracy between these models. Since we are interested in high predictive power and the radial kernel had the highest prediction accuracy, we chose this kernel.

Standardizing the predictor variables in SVM and encoding categorical variables has been shown to improve performance for SVMs (<https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>). Thus, we used the standardized forms of the continuous variables in our model and encoded the sex variable so that if a child

was a male we would code that as 1. We did indeed observe a small improvement in prediction accuracy across all models. However, one problem that particularly piqued our interests is that we are making no predictions for sites 4 or 5. This could be due to sites 4 & 5 having a smaller number of observations than the other classes. Thus, we used the class weight formula below to assign weights to each class and specify “one versus one” comparison, which has been suggested to yield better prediction than “one versus all.”

$$w_j = \frac{n}{kn_j}, \text{ n is total number of data points, k is number of classes, } n_j \text{ is the number of data in class j}$$

After the class weight assignment, the SVM models began to make prediction on class 4 & 5. However, doing so came at the cost of overall accuracy. Thus, more of the other observations are being misclassified, but the few observations of class 4&5 are correctly classified. To remedy this issue, we began to experiment with the class weights and increase the class 4 & 5 weights by roughly 0.5, which is arbitrarily chosen, and it boosted linear kernel SVM’s prediction accuracy to 0.403.