Group8-HW6

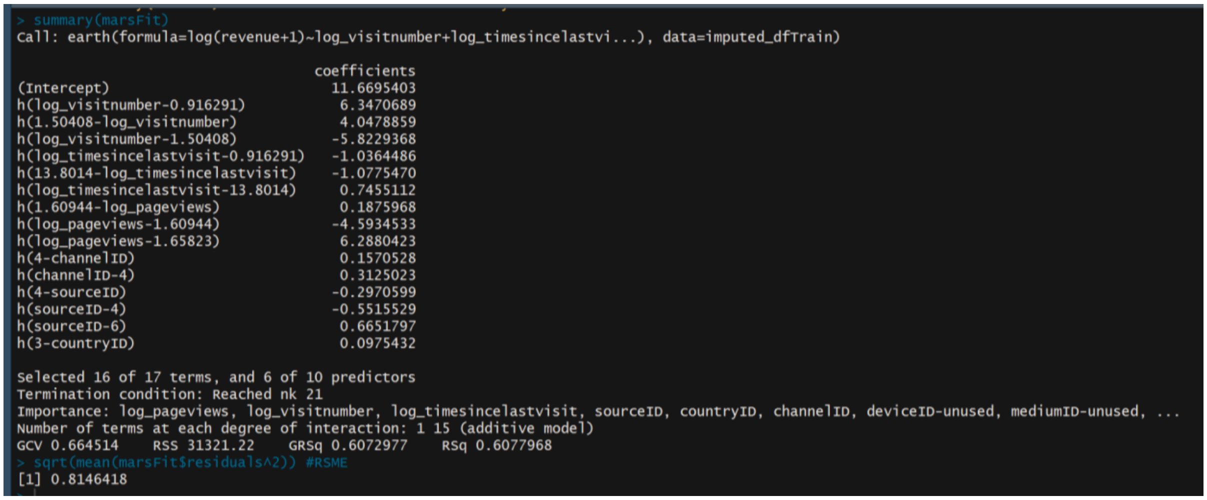
Group 8 – Lince Rumainum

October 26, 2019

Problem 1

Problem 1-(a)

Using MARS model from earth package, I tried to tune in the model by using the degree hyperparameter (coefficients and how the model behave for different hyperparameters are shown in **Figure 1** to **Figure 8**). At first with the default value degree of 1, it resulted in coefficients and RSME below (**Figure 1** and **Figure 2**):



```
> summary(marsFit)
Call: earth(formula=log(revenue+1)~log_visitnumber+log_timesincelastvi...), data=imputed_dfTrain)

                                   coefficients
(Intercept)                          11.6695403
h(log_visitnumber-0.916291)           6.3470689
h(1.50408-log_visitnumber)            4.0478859
h(log_visitnumber-1.50408)           -5.8229368
h(log_timesincelastvisit-0.916291)   -1.0364486
h(13.8014-log_timesincelastvisit)    -1.0775470
h(log_timesincelastvisit-13.8014)     0.7455112
h(1.60944-log_pageviews)              0.1875968
h(log_pageviews-1.60944)             -4.5934533
h(log_pageviews-1.65823)              6.2880423
h(4-channelID)                        0.1570528
h(channelID-4)                        0.3125023
h(4-sourceID)                        -0.2970599
h(sourceID-4)                        -0.5515529
h(sourceID-6)                         0.6651797
h(3-countryID)                        0.0975432

Selected 16 of 17 terms, and 6 of 10 predictors
Termination condition: Reached nk 21
Importance: log_pageviews, log_visitnumber, log_timesincelastvisit, sourceID, countryID, channelID, deviceID-unused, mediumID-unused, ...
Number of terms at each degree of interaction: 1 15 (additive model)
GCV 0.664514    RSS 31321.22    GRSq 0.6072977    RSq 0.6077968
> sqrt(mean(marsFit$residuals^2)) #RSME
[1] 0.8146418
```

Figure 1 *MARS Model with Degree Hyperparameter of 1*

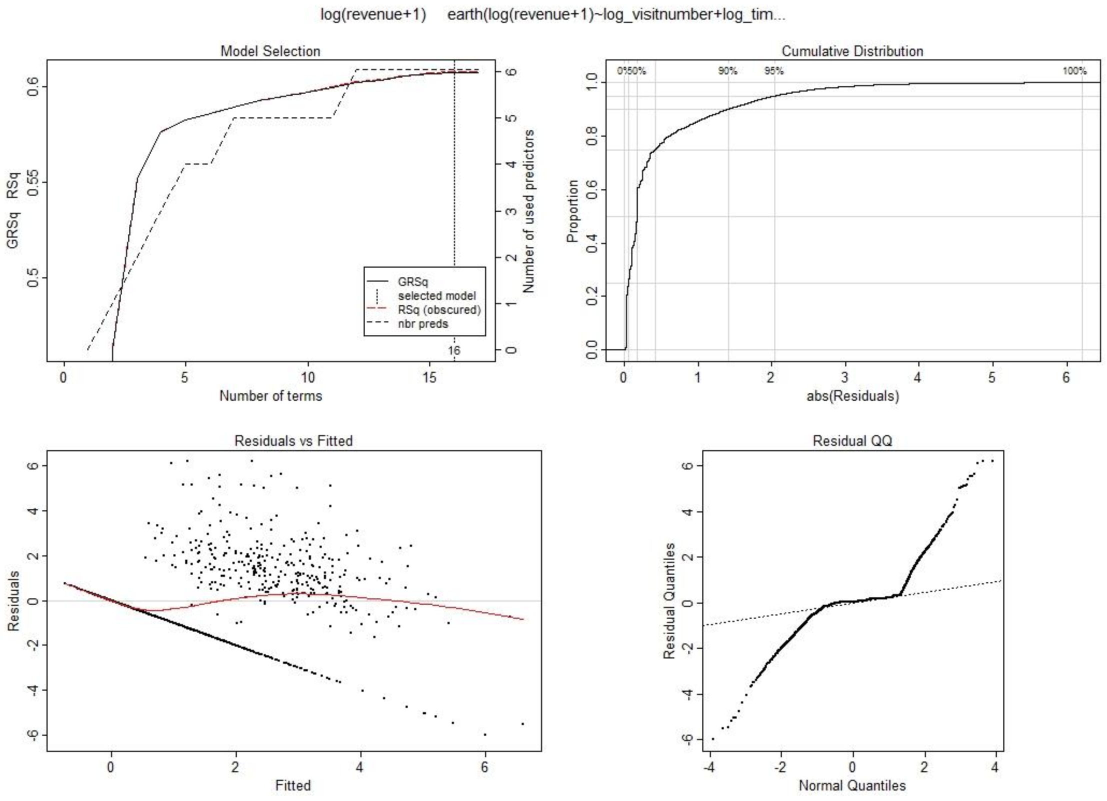
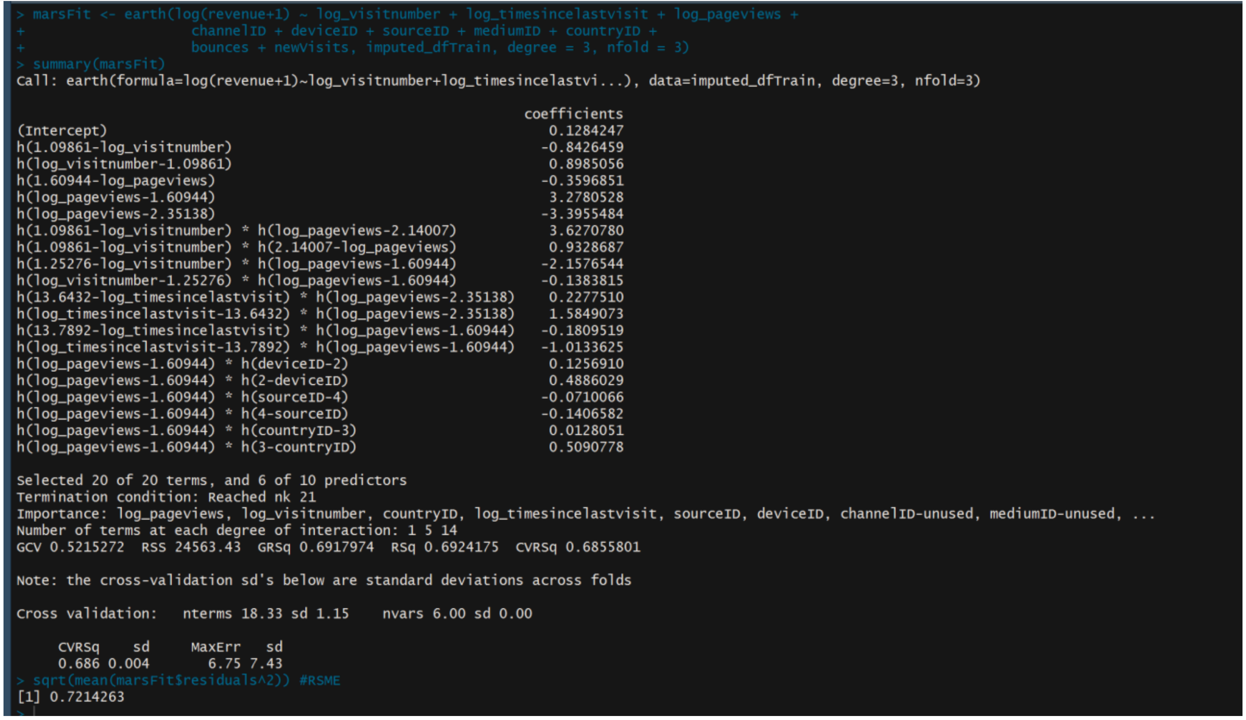


Figure 2 *Plot of MARS Model with Degree Hyperparameter of 1*

When the degree parameters change to 3, the results improved to RSME = 0.7214 with coefficients shown below in **Figure 3** and **Figure 4**:



```
> marsFit <- earth(log(revenue+1) ~ log_visitnumber + log_timesincelastvisit + log_pageviews +
+                    channelID + deviceID + sourceID + mediumID + countryID +
+                    bounces + newVisits, imputed_dfTrain, degree = 3, nfold = 3)
> summary(marsFit)
Call: earth(formula=log(revenue+1)~log_visitnumber+log_timesincelastvi...), data=imputed_dfTrain, degree=3, nfold=3)

                                                         coefficients
(Intercept)                                                 0.1284247
h(1.09861-log_visitnumber)                                 -0.8426459
h(log_visitnumber-1.09861)                                  0.8985056
h(1.60944-log_pageviews)                                   -0.3596851
h(log_pageviews-1.60944)                                    3.2780528
h(log_pageviews-2.35138)                                   -3.3955484
h(1.09861-log_visitnumber) * h(log_pageviews-2.14007)       3.6270780
h(1.09861-log_visitnumber) * h(2.14007-log_pageviews)       0.9328687
h(1.25276-log_visitnumber) * h(log_pageviews-1.60944)      -2.1576544
h(log_visitnumber-1.25276) * h(log_pageviews-1.60944)      -0.1383815
h(13.6432-log_timesincelastvisit) * h(log_pageviews-2.35138)     0.2277510
h(log_timesincelastvisit-13.6432) * h(log_pageviews-2.35138)     1.5849073
h(13.7892-log_timesincelastvisit) * h(log_pageviews-1.60944)    -0.1809519
h(log_timesincelastvisit-13.7892) * h(log_pageviews-1.60944)    -1.0133625
h(log_pageviews-1.60944) * h(deviceID-2)                    0.1256910
h(log_pageviews-1.60944) * h(2-deviceID)                    0.4886029
h(log_pageviews-1.60944) * h(sourceID-4)                   -0.0710066
h(log_pageviews-1.60944) * h(4-sourceID)                   -0.1406582
h(log_pageviews-1.60944) * h(countryID-3)                   0.0128051
h(log_pageviews-1.60944) * h(3-countryID)                   0.5090778

Selected 20 of 20 terms, and 6 of 10 predictors
Termination condition: Reached nk 21
Importance: log_pageviews, log_visitnumber, countryID, log_timesincelastvisit, sourceID, deviceID, channelID-unused, mediumID-unused, ...
Number of terms at each degree of interaction: 1 5 14
GCV 0.5215272  RSS 24563.43  GRSq 0.6917974  RSq 0.6924175  CVRSq 0.6855801

Note: the cross-validation sd's below are standard deviations across folds

Cross validation:   nterms 18.33 sd 1.15    nvars 6.00 sd 0.00

     CVRSq    sd     MaxErr   sd
     0.686 0.004       6.75 7.43
> sqrt(mean(marsFit$residuals^2)) #RSME
[1] 0.7214263
```

Figure 3 *MARS Model with Degree Hyperparameter of 3 and 3 number of folds*

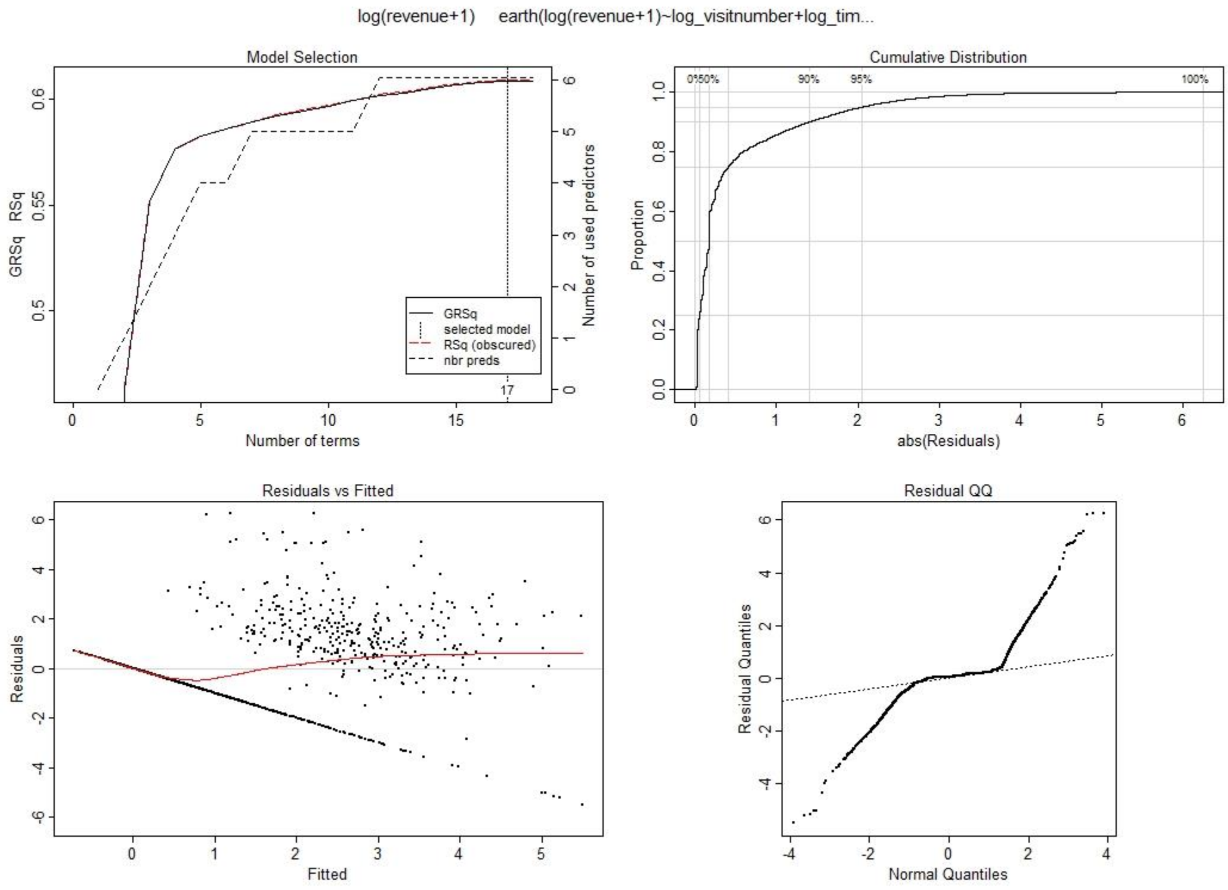
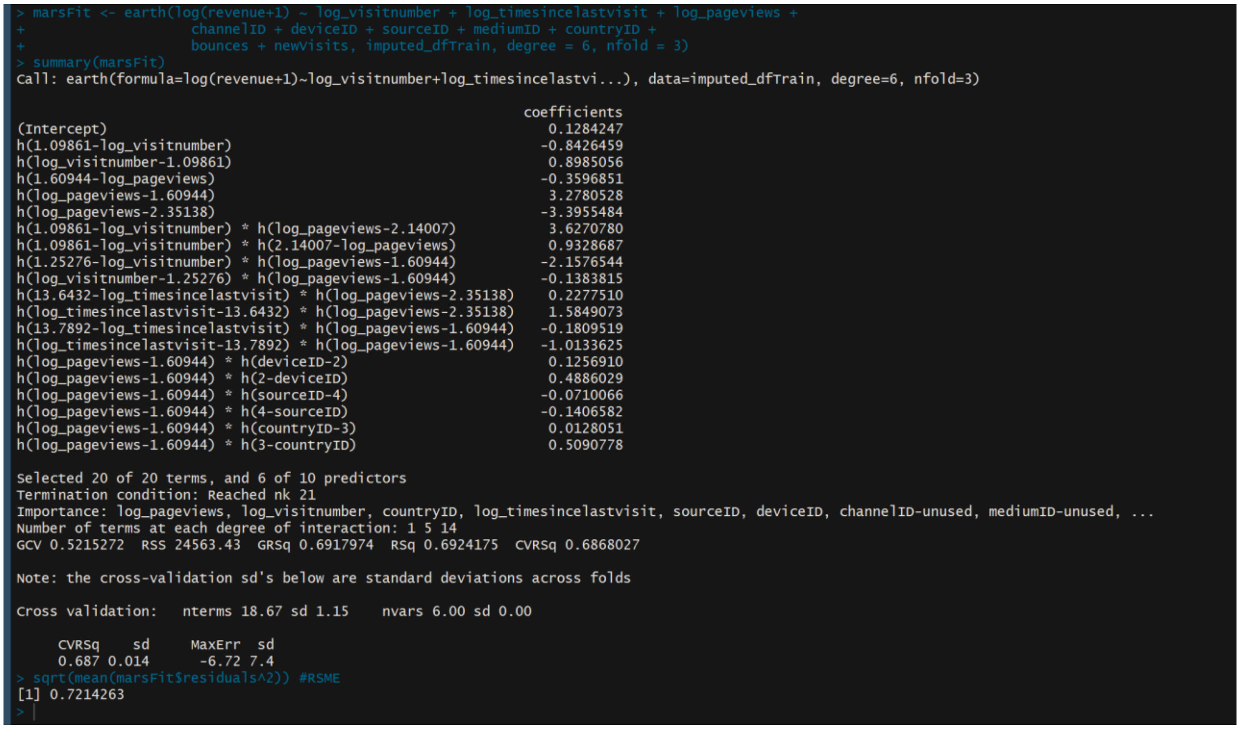


Figure 4 *Plot of MARS Model with Degree Hyperparameter of 3 and 3 number of folds*

When the degree parameter and/or the fold increases to 6, the results did not improve and stays the same as shown below:



```
> marsFit <- earth(log(revenue+1) ~ log_visitnumber + log_timesincelastvisit + log_pageviews +
+                   channelID + deviceID + sourceID + mediumID + countryID +
+                   bounces + newVisits, imputed_dfTrain, degree = 6, nfold = 3)
> summary(marsFit)
Call: earth(formula=log(revenue+1)~log_visitnumber+log_timesincelastvi...), data=imputed_dfTrain, degree=6, nfold=3)

                                                           coefficients
(Intercept)                                                   0.1284247
h(1.09861-log_visitnumber)                                   -0.8426459
h(log_visitnumber-1.09861)                                    0.8985056
h(1.60944-log_pageviews)                                     -0.3596851
h(log_pageviews-1.60944)                                      3.2780528
h(log_pageviews-2.35138)                                     -3.3955484
h(1.09861-log_visitnumber) * h(log_pageviews-2.14007)         3.6270780
h(1.09861-log_visitnumber) * h(2.14007-log_pageviews)         0.9328687
h(1.25276-log_visitnumber) * h(log_pageviews-1.60944)        -2.1576544
h(log_visitnumber-1.25276) * h(log_pageviews-1.60944)        -0.1383815
h(13.6432-log_timesincelastvisit) * h(log_pageviews-2.35138)  0.2277510
h(log_timesincelastvisit-13.6432) * h(log_pageviews-2.35138)  1.5849073
h(13.7892-log_timesincelastvisit) * h(log_pageviews-1.60944) -0.1809519
h(log_timesincelastvisit-13.7892) * h(log_pageviews-1.60944) -1.0133625
h(log_pageviews-1.60944) * h(deviceID-2)                      0.1256910
h(log_pageviews-1.60944) * h(2-deviceID)                      0.4886029
h(log_pageviews-1.60944) * h(sourceID-4)                     -0.0710066
h(log_pageviews-1.60944) * h(4-sourceID)                     -0.1406582
h(log_pageviews-1.60944) * h(countryID-3)                     0.0128051
h(log_pageviews-1.60944) * h(3-countryID)                     0.5090778

Selected 20 of 20 terms, and 6 of 10 predictors
Termination condition: Reached nk 21
Importance: log_pageviews, log_visitnumber, countryID, log_timesincelastvisit, sourceID, deviceID, channelID-unused, mediumID-unused, ...
Number of terms at each degree of interaction: 1 5 14
GCV 0.5215272  RSS 24563.43  GRSq 0.6917974  RSq 0.6924175  CVRSq 0.6868027

Note: the cross-validation sd's below are standard deviations across folds

Cross validation:   nterms 18.67 sd 1.15    nvars 6.00 sd 0.00

     CVRSq    sd     MaxErr  sd
     0.687 0.014      -6.72 7.4
> sqrt(mean(marsFit$residuals^2)) #RSME
[1] 0.7214263
> 
```

Figure 5 *MARS Model with Degree Hyperparameter of 6 and 3 number of folds*

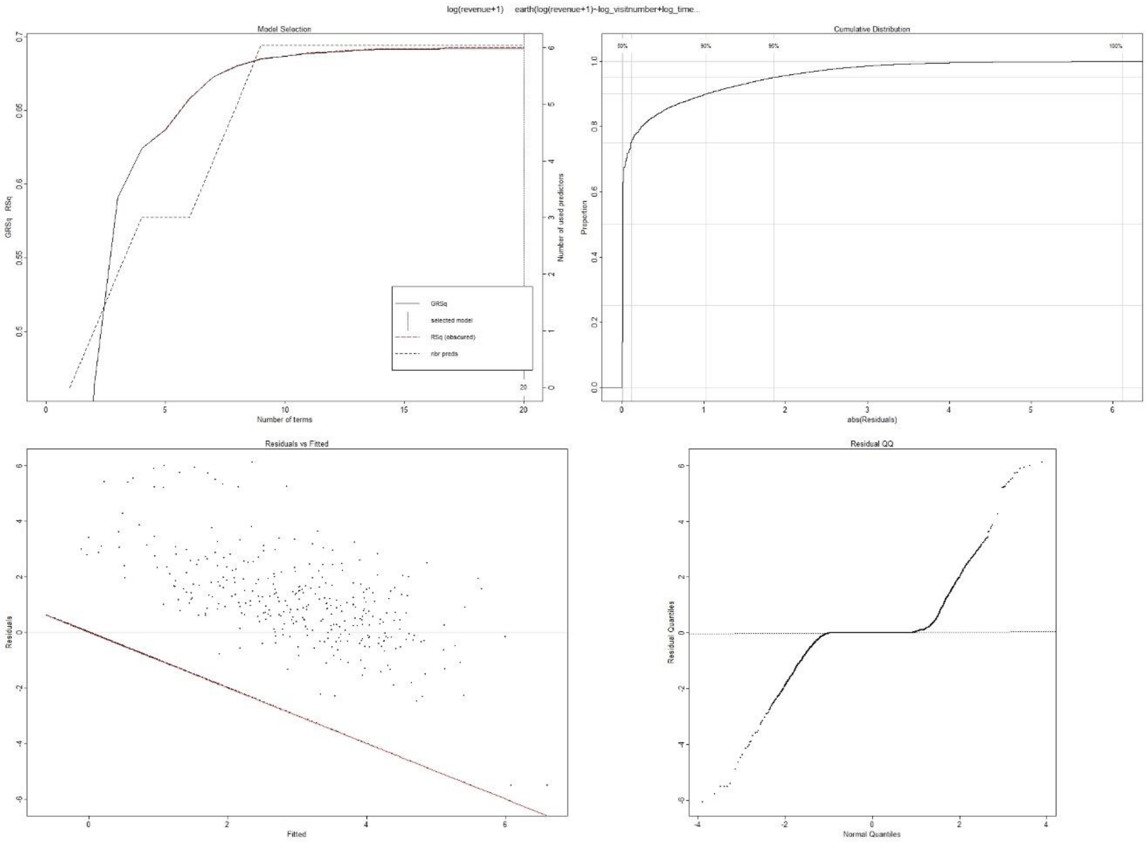
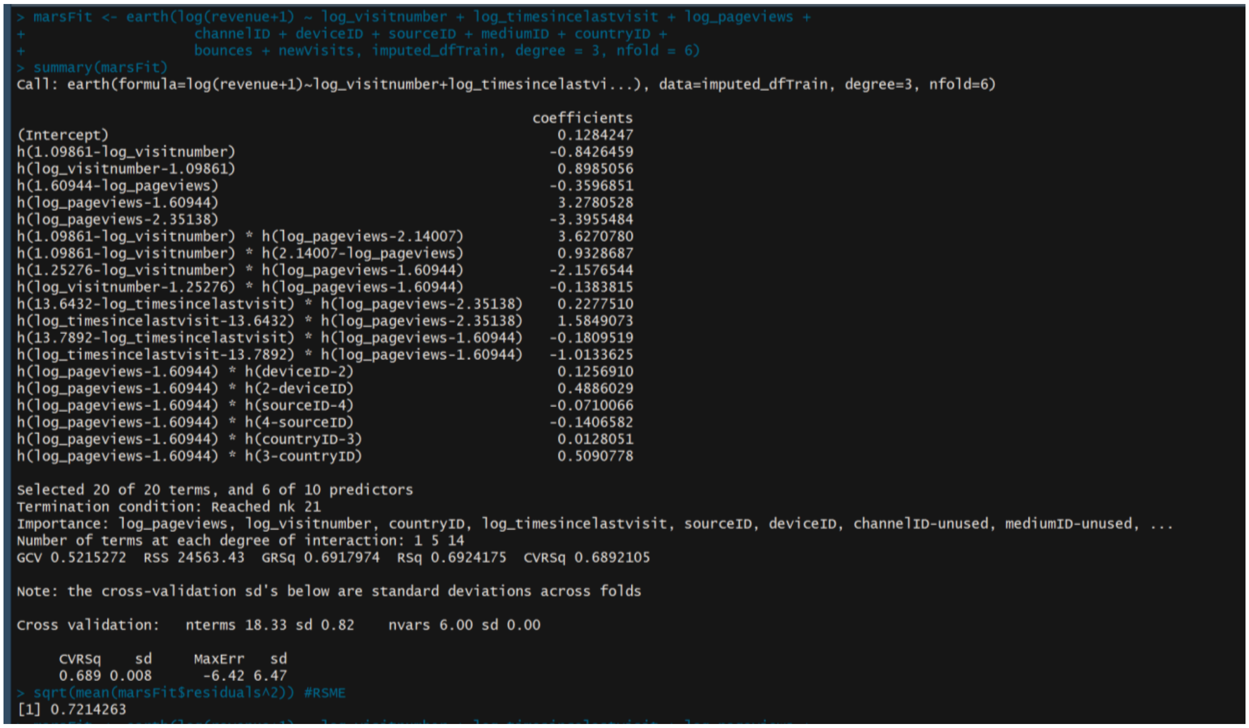


Figure 6 *Plot of MARS Model with Degree Hyperparameter of 6 and 3 number of folds*



```
> marsFit <- earth(log(revenue+1) ~ log_visitnumber + log_timesincelastvisit + log_pageviews +
+                    channelID + deviceID + sourceID + mediumID + countryID +
+                    bounces + newVisits, imputed_dfTrain, degree = 3, nfold = 6)
> summary(marsFit)
Call: earth(formula=log(revenue+1)~log_visitnumber+log_timesincelastvi...), data=imputed_dfTrain, degree=3, nfold=6)

                                                             coefficients
(Intercept)                                                     0.1284247
h(1.09861-log_visitnumber)                                     -0.8426459
h(log_visitnumber-1.09861)                                      0.8985056
h(1.60944-log_pageviews)                                       -0.3596851
h(log_pageviews-1.60944)                                        3.2780528
h(log_pageviews-2.35138)                                       -3.3955484
h(1.09861-log_visitnumber) * h(log_pageviews-2.14007)           3.6270780
h(1.09861-log_visitnumber) * h(2.14007-log_pageviews)           0.9328687
h(1.25276-log_visitnumber) * h(log_pageviews-1.60944)          -2.1576544
h(log_visitnumber-1.25276) * h(log_pageviews-1.60944)          -0.1383815
h(13.6432-log_timesincelastvisit) * h(log_pageviews-2.35138)    0.2277510
h(log_timesincelastvisit-13.6432) * h(log_pageviews-2.35138)    1.5849073
h(13.7892-log_timesincelastvisit) * h(log_pageviews-1.60944)   -0.1809519
h(log_timesincelastvisit-13.7892) * h(log_pageviews-1.60944)   -1.0133625
h(log_pageviews-1.60944) * h(deviceID-2)                        0.1256910
h(log_pageviews-1.60944) * h(2-deviceID)                        0.4886029
h(log_pageviews-1.60944) * h(sourceID-4)                       -0.0710066
h(log_pageviews-1.60944) * h(4-sourceID)                       -0.1406582
h(log_pageviews-1.60944) * h(countryID-3)                       0.0128051
h(log_pageviews-1.60944) * h(3-countryID)                       0.5090778

Selected 20 of 20 terms, and 6 of 10 predictors
Termination condition: Reached nk 21
Importance: log_pageviews, log_visitnumber, countryID, log_timesincelastvisit, sourceID, deviceID, channelID-unused, mediumID-unused, ...
Number of terms at each degree of interaction: 1 5 14
GCV 0.5215272  RSS 24563.43  GRSq 0.6917974  RSq 0.6924175  CVRSq 0.6892105

Note: the cross-validation sd's below are standard deviations across folds

Cross validation:   nterms 18.33 sd 0.82    nvars 6.00 sd 0.00

     CVRSq    sd     MaxErr   sd
     0.689 0.008      -6.42 6.47
> sqrt(mean(marsFit$residuals^2)) #RSME
[1] 0.7214263
```

Figure 7 *MARS Model with Degree Hyperparameter of 3 and 6 number of folds*

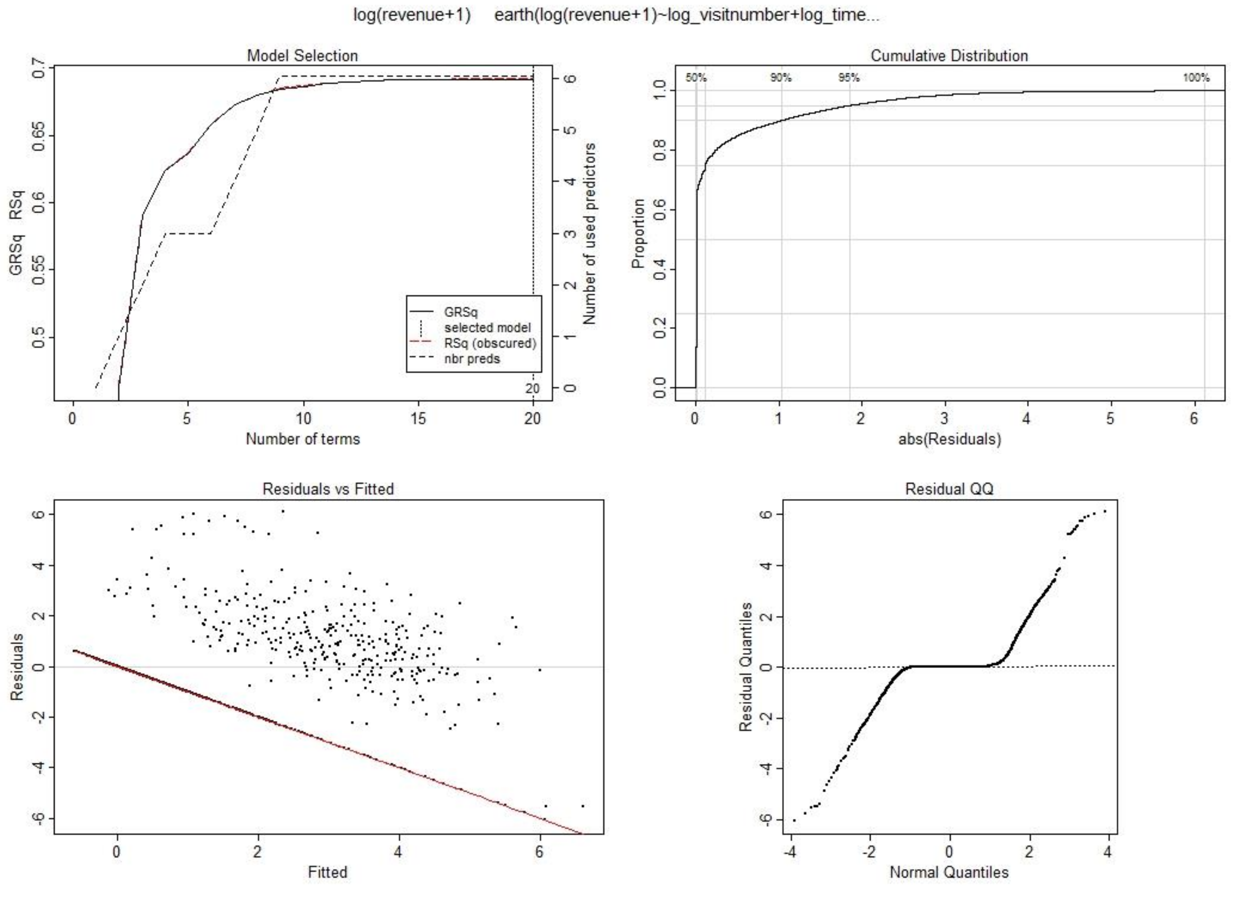


Figure 8 *Plot of MARS Model with Degree Hyperparameter of 3 and 6 number of folds*

For different model, such as, ridge and lasso, the lambdas that would give the minimum mean-squared error was calculated with glmnet package function and **Figure 9** below are the graph how coefficients and predictors behave for different lambda values and how the mean-squared error and predictors behave for different lambda values:

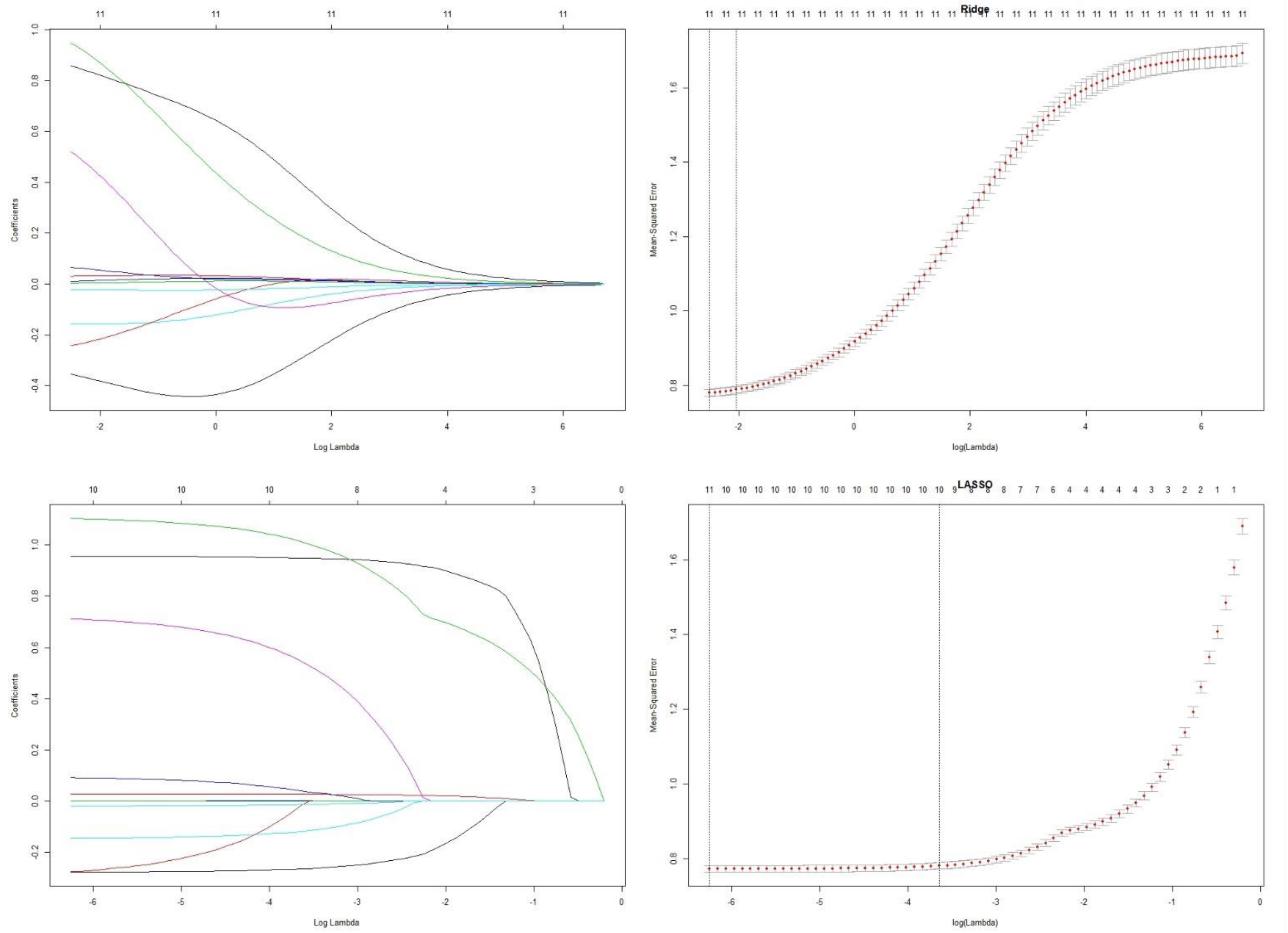
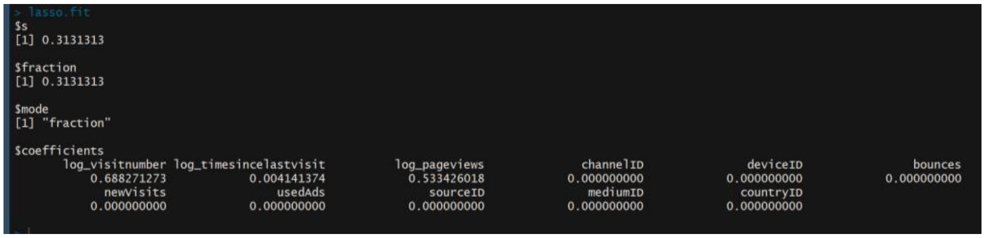


Figure 9 *Plot of Ridge (Top) and Lasso (Bottom) Model Behavior with Different Lambda Values*

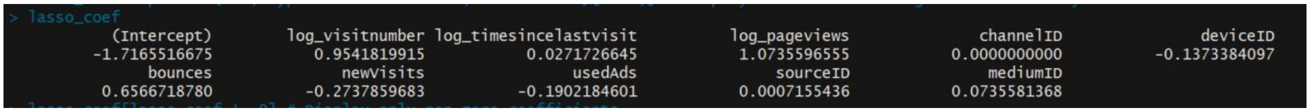
Other than using MARS model, I also use the lasso model. I used one of the lasso models with fraction hyperparameter of **elasticnet** package while the other one used the **gmlnet** package where it selection came from the minimum lambda value. For the first model, using the fractions hyperparameter for the lasso, the **Figure 10** below shows that *log_visitnumber*, *log_timesincelastvisit*, and *log_pageviews* are the only non-zero coefficients:



```
> lasso.fit
$s
[1] 0.3131313

$fraction
[1] 0.3131313

$mode
[1] "fraction"

$coefficients
      log_visitnumber log_timesincelastvisit          log_pageviews              channelID               deviceID                bounces
          0.688271273            0.004141374            0.533426018            0.000000000            0.000000000            0.000000000
            newVisits                usedAds               sourceID               mediumID              countryID
          0.000000000            0.000000000            0.000000000            0.000000000            0.000000000
```

Figure 10 *Variable Coefficients using Lasso's Elasticnet Package*

While using the glmnet package the only zero coefficient is *channelID*, as shown on the **Figure 11** below:



```
> lasso_coef
         (Intercept)       log_visitnumber log_timesincelastvisit         log_pageviews             channelID              deviceID
       -1.7165516675          0.9541819915           0.0271726645          1.0735596555          0.0000000000         -0.1373384097
             bounces             newVisits                usedAds              sourceID              mediumID
        0.6566718780         -0.2737859683          -0.1902184601          0.0007155436          0.0735581368
```

Figure 11 *Variable Coefficients using Lasso's glmnet Package*

The lasso with fraction hyperparameter give 0.31313 as its selection while the lasso with lambda hyperparameter is 0.0002915053. From the previous figures with the coefficients, it shows that other than both channelID variable having zero value coefficient, the other coefficients are different from each other.

Here are all data of the five non-linear models and the linear model from HW-5:

Table 1 *R2 and RSME Data for Different Models (Linear and Non-Linear)*

					CV performance	
Model	Method	Package	Hyperparameter	Selection	R^2	RMSE
OLS HW-5	lm	stats	NA	NA	0.5426	0.8798
Lasso	lasso	elasticnet	fraction	0.31313		
Lasso	lasso	glmnet	lambda	0.00029		
Ridge	ridge	glmnet	lambda	0.00029		0.8723
Huber loss	rlm	MASS	NA	NA		1.1623
MARS	earth	earth	degree	3	0.6924	0.7214

Problem 1-(b)

The best model I used was by using the MARS method. As discussed in the beginning of part (a), I used different hyperparameter to see if one is better than the others and found that hyperparameter with degree of three gives the best cross validation performance. I chose attributes that can be categorized and create a numerical factor representation for them, create a feature regarding ads, and the three numeric data fields, which are *newVisits*, *timeSinceLastVisit*, and *pageviews*. I did not considere date at the moment because I would like to approach the problem with all the attributes and I think that adding date might cause the linear model to overfit. I created several new categorized attributes with numerical IDs. Those new features are *channelID*, *deviceID*, *sourceID*, *mediumID*, *countryID*. By using the fct_lump function from **forcats** package, I was able to create factors for channel, device, source, medium, and country column, where 1 is the most occurrences for that category and 2 the next most number of occurrences and so on.

- For channel, there are eight categories: 1-Organic Search, 2-Social, 3-Referral, 4-Direct, 5-Paid Search, 6-Affiliates, 7-Display, and 8-(Other).
- For device, there are three categories: 1-desktop, 2-mobile, and 3-tablet.
- For source, there are ten categories: 1-“google”, 2-“youtube.com”, 3-“(direct), 4-mall.googleplex.com, 5-analytics.google.com, 6-Partners, 7-dfa, 8-google.com, 9-sites.google.com, 10-other.
- For medium there five categories: 1-organic, 2-referral, 3-cpc, 4-affiliate, 5-cpm.

- For country there ten categories: 1-United States, 2-India, 3-United Kingdom, 4-Canada, 5-Vietnam, 6-Thailand, 7-Turkey, 8-Germany, 9-Brazil, 10-Other.

I also add a column that indicates if ads were part of it and impute the NA values for bounces and newVisits variables to zeroes since those indicates that it did not bounce and it was not their first visit.

For the MARS model itself, the tuning I did were from the number of folds and degree parameters. For details of the MARS model's data with the best result refer to **Figure 3** and **Figure 4**.