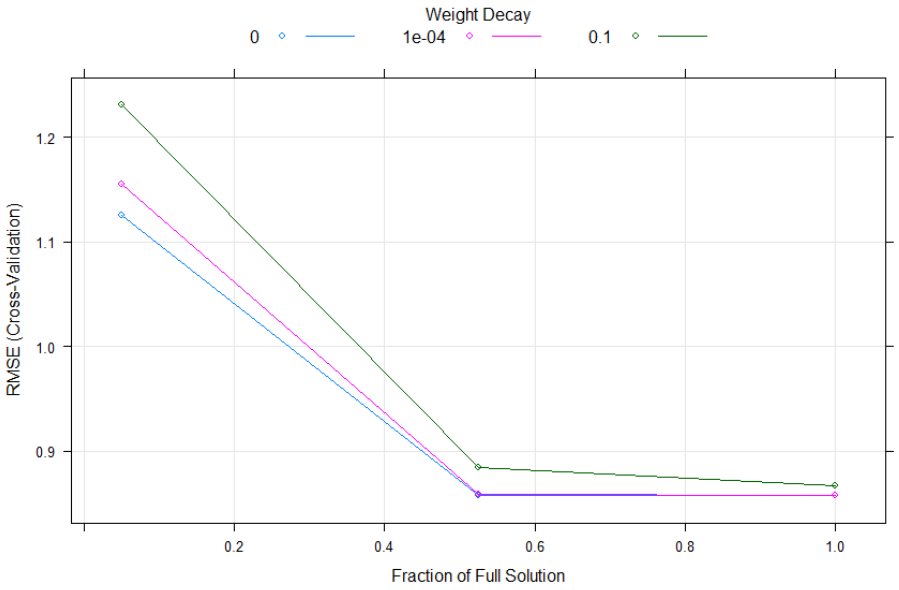
Brian Keafer HW#6

Sales Prediction Problem

Part (A)

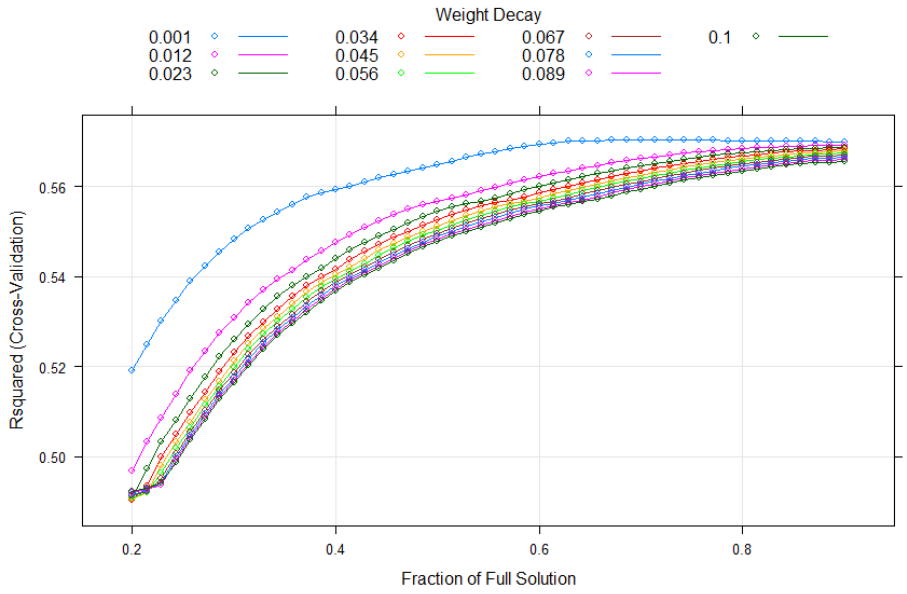
Section I – Hyperparameter Justification on Elasticnet Model

The Elasticnet regression model displays the most accurate, intuitive, and most easily displayed representation of the optimization techniques used on the hyperparameters, lambda and fraction. The initial creation of the model contained a generic sequence that provided a baseline model to improve on. This generic model (seen to the right), that lacked the inclusion of the "expand.grid" function, optimized the hyperparameters as follows, fraction = 1 and lambda = 0. However, with the use of "expand.grid" function we are able to optimize both of these hyperparameters to a more detailed level. An example can be seen below:

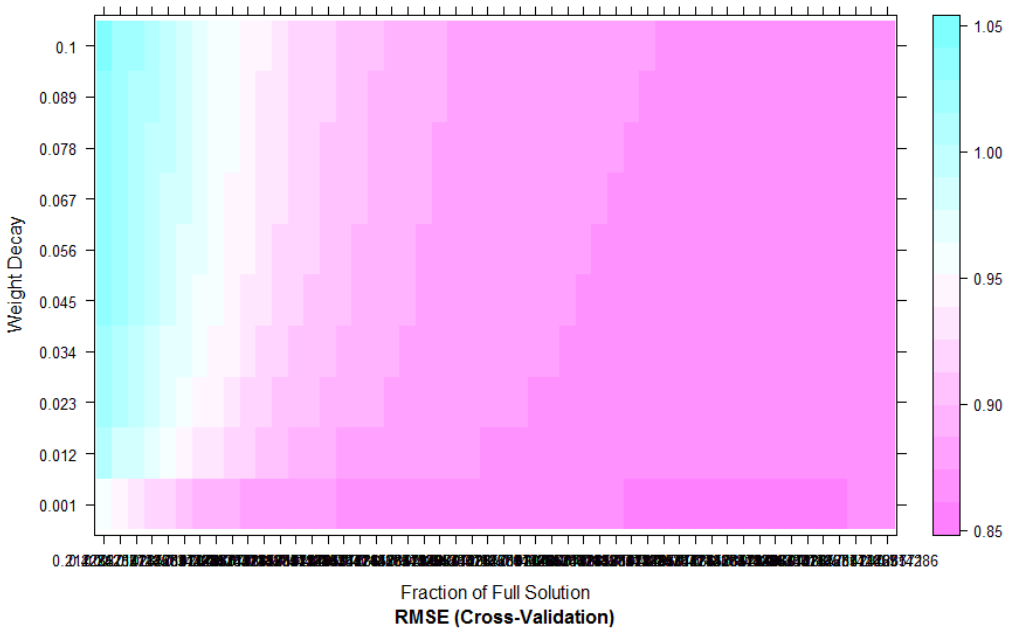


enetGrid <- expand.grid(lambda=seq(0.001,0.1,length=10), fraction=seq(.2, .9, length = 50))

The expansion of the grid in which the model conducts optimization techniques is expanded by the use of this function and it is referenced as an argument in the "train" function. Many iterations of tuning the hyperparameters was needed in order to help the Elasticnet model identify the most optimal hyperparameters, lambda = 0.001 and fraction = 0.7285714. The two visualizations of optimization on the right support this argument. The first compares the re-sampled r-squared value against the fraction parameter. It is very clear all fraction instances have a diminishing return on the r-squared value.



The plot to the right is the clearest graphical representation of the optimization of both hyperparameters simultaneously. The darkest area on the graph represent the area where the lowest RMSE Cross-Validation score is possible. This clearly matches the optimized levels the Elasticnet model produced.



Section II – Compare/Contrast PLS Model and MARS Model

A MARS model and the PLS model are different in their very nature. Partial Least Squares (PLS) is similar to PCA, as we remove components of our data. Optimizing on the hyper-parameter “n-comp”. PLS takes many iterations to identify the most optimal component. This takes a decent amount of computing powers. MARS diverges away from the dimension reduction behaviors present in PLS, MARS asses every combination due to hyperparameter “degree”. Additionally, “nprune” functions to determine the number of terms to keep. The coefficient variables in each model are essentially the same in order to preserve a pure comparison between these two models. While both models require a fair bit of computing power, the usefulness of each is very different. It is quite clear that MARS outperformed PLS in cross validation performance, and in fact was the top performing model overall. This can be seen in section III comparisons.

Section III – Model Performances

Model	Method	Package	Hyperparameter	Selection	R^2	RMSE
OLS - HW-6	lm	stats	NA	NA	0.57	0.861
lasso	lasso	elasticnet	fraction	0.759	0.57	0.859
ridge	ridge	elastinet	lambda	0.001	0.572	0.858
elastinet	enet	elastinet	lambda, fraction	0.001, 0.8	0.573	0.858
Partial Least Squares	PLS	pls	ncomp	23	0.57	0.86
MARS	earth	earth	degree, nprune	1,16	0.675	0.746

Part (B)

Section I – Best Regression Model

The most successful regression model produced was the MARS model. The re-sampled R-squared value is 0.675 and the re-sampled RSME is 0.746. The tuning parameter that affected the model included “degree” and “nprune”. The optimized values were degree = 1 and nprune = 16. My modeling approach specifically concentrated upon iterations on the optimization steps, concentrating on the hyperparameter values. Additionally, the removal of variables that lacked any significant variance via the “nearZeroVar”, allowed the model to work in a more accurate way. The initial transformations made to the train and test data sets paid dividends in this model, because it removed collinearity from the training data. Additionally, breaking down the factor variables to more condensed categories created more significant relationships within the data. MARS is able to prioritize the terms applied to the model and execute them accordingly. The preciseness of my transformations and addition of some numeric features allowed for increased accuracy within the model. Further, the use of the function “expand.grid” allowed for optimization techniques to be applied fairly easily. Precise data preparation work was a driving force behind the success of this model.

The “summary” function was applied to the most accurate MARS model. The results are displayed below and includes variables, coefficients, and p-values.

```
> summary(MARS_fit)
Call: earth(x=matrix[47249,43], y=c(0,0,0,0,0,0,0...), keepxy=TRUE, degree=1, nprune=16)

                          coefficients
(Intercept)                 -3.9476179
operatingSystemMacintosh     0.1024357
countryOther                -0.1470029
regionNew York               0.2450623
sourcemall.googleplex.com    0.3965546
h(bounce_sessions-16)        0.0577282
h(pageviews_sum-9)           0.1545543
h(47-pageviews_sum)          0.0993042
h(pageviews_sum-47)         -0.1404320
h(pageviews_sum-375)        -0.0102415
h(5-pageviews_max)          -0.0687647
h(pageviews_max-5)           0.1252246
h(pageviews_max-16)         -0.0804916
h(pageviews_max-24)         -0.0537076
h(3-session_cnt)            -0.1299801
h(session_cnt-3)            -0.0651311

Selected 16 of 19 terms, and 8 of 43 predictors
Termination condition: RSq changed by less than 0.001 at 19 terms
Importance: pageviews_sum, pageviews_max, sourcemall.googleplex.com, countryOther, sessio
n_cnt, ...
Number of terms at each degree of interaction: 1 15 (additive model)
GCV 0.5468613    RSS 25804.75    GRSq 0.6811294    RSq 0.6815342
```