STATS 503 HOMEWORK 4

March 17, 2018

1 OVERVIEW

We use an email spam dataset to predict whether or not emails are spam based on 57 features. This dataset has over 3,000 observations to train on and over 1,000 to predict on.

We classify spam emails through support vector machines (SVM), neural networks, and decision trees. We first choose to standardize our data so the predicted values of the SVM/neural network/decision tree can be accurately compared to that of the actual data. For neural network in particular failure to normalize the data often results in the prediction value remaining the same across all observations regardless of the input.

We will examine our results through 10-fold cross validation for SVM, neural networks, and decision trees. After choosing our optimal model we apply the final models for each and view the train and test error.

Throughout this we also examine our given dataset (which we call the Balanced, referring to the balanced sample), before we adjust the ratios to view more skewed class ratios, specifically 3:7, 2:8, and 1:9. Finally, we bootstrap data to see if this improves the classification performance on each of the three methods.

2 SVM

We examine support vector machines as a method of classification. For SVM we use the tuning parameter cost which is the weight for penalizing the soft margin. We test costs from exponential -2 to 2, which acts as a control over the total amount of slack allowed.

We also examine the tuning parameters for certain kernel choices, of which we use Linear, Gaussian, and Polynomial. Our tuning parameters for Linear are just costs. For Gaussian we vary cost and gamma (.01, .05, .1 respectively). A small gamma acts as a distribution with large variance. Therefore the support vector has an influence on deciding the class of a given point even if the distance between them is large. Conversely, for a large gamma, the support vector does not have wide spread influence on determining the class of a given point. Finally, for Polynomials we adjust cost, gamma, and degrees (quadratic and cubic).

We optimize over these different parameters, cross-validating for each combination, and see the averaged following results. Below is how we would interpret our SVM graphs based on our balanced data, and we will then focus only on our optimal ones.

We compute our final train and test errors on each of the best model by kernel type. We choose our train/test based on our optimal model which produced the lowest cross-validated error rate, by kernel type. We then compute final train and test errors to see which of our optimal model produces the best results compared to the other kernels.

Interestingly we see our cross-validated errors for the most part are very steady past a cost of 2 for every kernel, aside from Polynomial with a Gamma of .01. It seems once a certain cost is reached our error remains fairly steady. In terms of gamma, when applicable, there is the most difference moving from no cost to a low cost penalty (aside from Polynomial with

gamma of 0.01), which makes sense because some amount of cost penalty for slackness seems better than none.

For degrees quadratic has much lower error for a given cost than cubic, but this goes away when there becomes a gamma of .05.

Finally we see the Gaussian optimal kernel has the lowest test error of 5.21 percent.

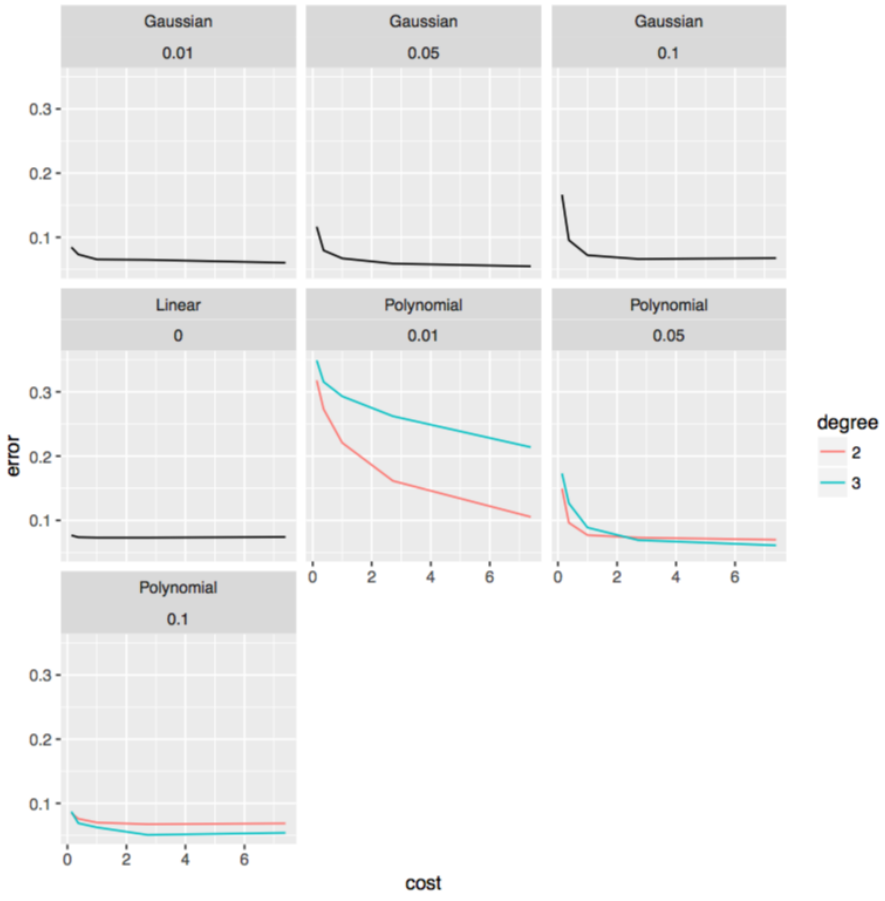
We now look across our different ratios to see if changing the ratio of spam and non-spam impacts our results, as well as resampling with bootstrapping.

We notice our other ratios still have close to the same error rate past a certain cost, but there is a much more steep tail in that low costs have very high error rates for these models. This seems impacted by the higher amounts of non-spam which at a low to no-cost (so little to no penalty) generates higher error amounts, which makes sense.

As with our normal model we notice our training errors are often extremely close to or 0, while our optimal test errors range from 5 to 14 percent. Our kernel type also varies in terms of which produces some optimal parameters for the best train and test. Our 3-7 ratio is Gaussian, our 2-8 is Linear, our 1-9 Polynomial, and resampled Linear.

Overall, our lowest error is for our original balanced dataset, which makes sense because we can sample equally and have an easier time correctly classifying our results. Otherwise our 3-7 does next best, followed by our resampled, then our 2-8, and finally our 1-9, all of which are consistent with the idea of balancing being very important in impacting our ability to correctly classify.

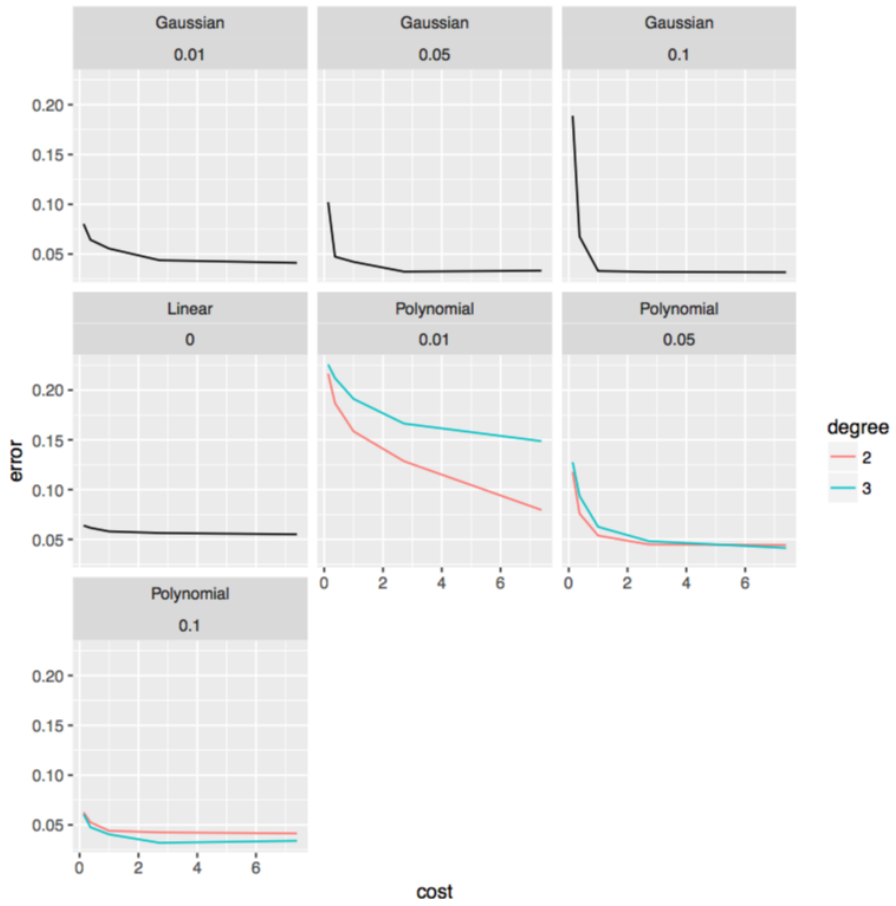
Balanced Cross-Validated SVM plots



Train/Test Best Error Balanced

	Test Error	Train Error	Type
1	0.0521512385919166	0.0123899576133029	Gaussian
2	0.0684485006518905	0.0619497880665145	Linear
3	0.0560625814863103	0.0120639060971634	Polynomial

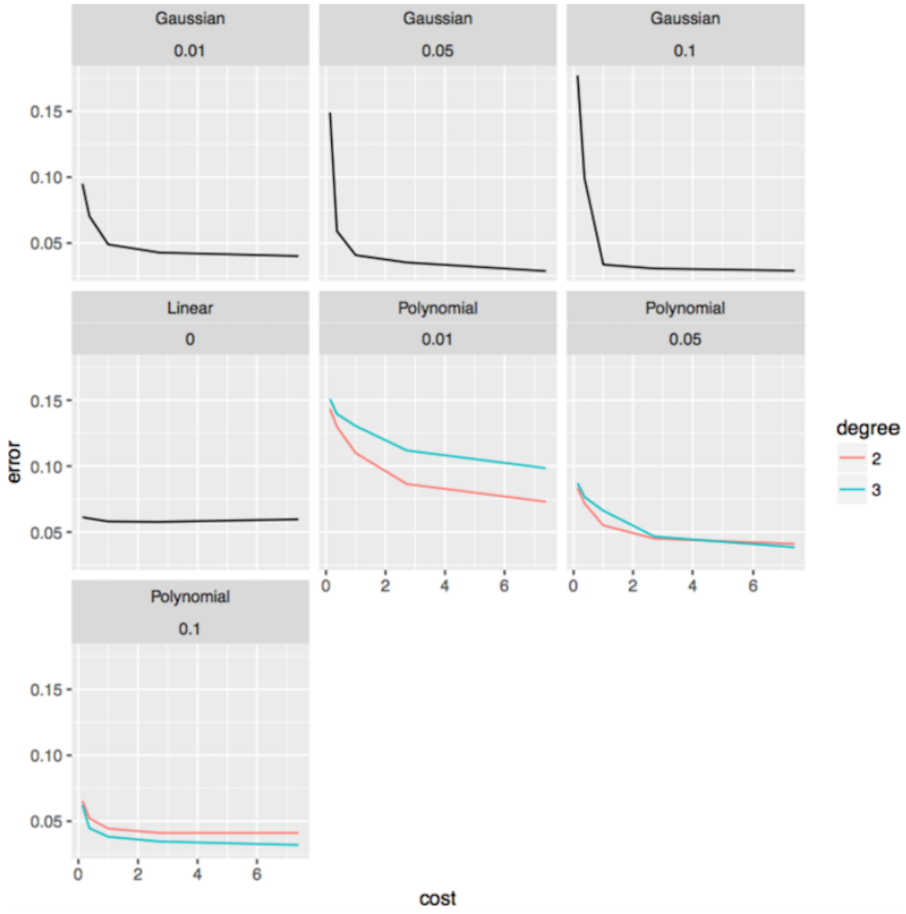
3-7 Ratio Cross-Validated SVM plots



Train/Test Best Error 3-7 Ratio

	Test Error	Train Error	Type
1	0.0710560625814863	0.00684931506849315	Gaussian
2	0.0775749674054759	0.0489236790606654	Linear
3	0.0893089960886571	0.0045662100456621	Polynomial

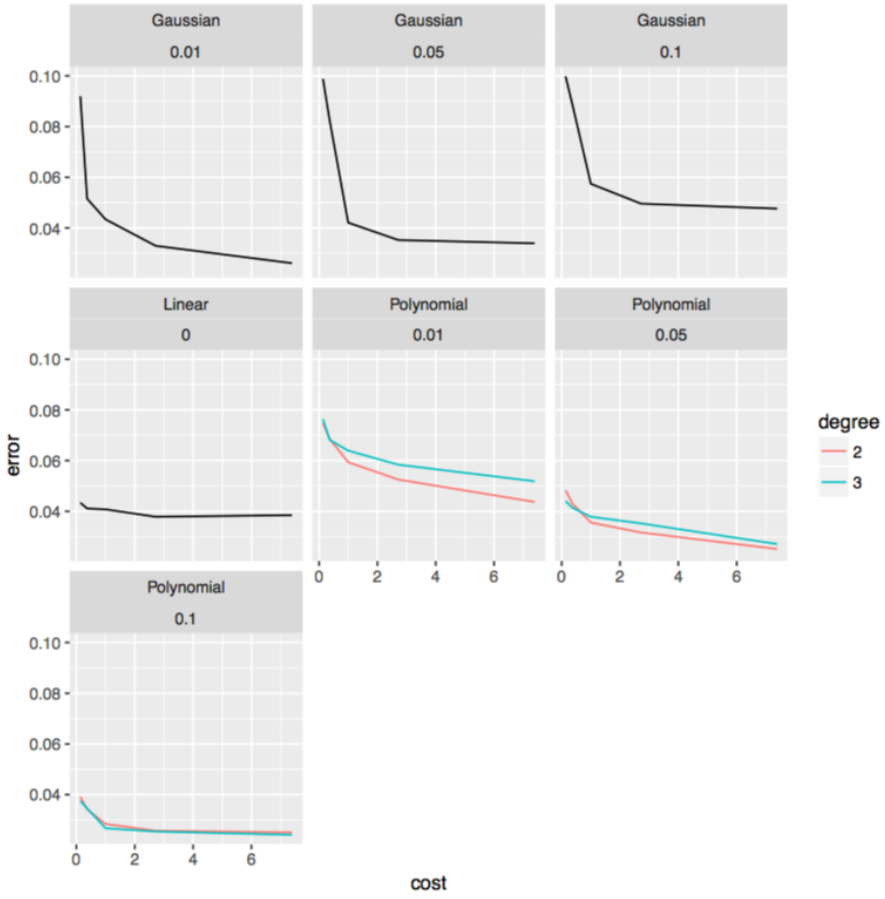
2-8 Ratio Cross-Validated SVM plots



Train/Test Best Error 2-8 Ratio

	Test Error	Train Error	Type
1	0.0899608865710561	0.00554468362687541	Gaussian
2	0.0886571056062582	0.0424005218525766	Linear
3	0.113428943937419	0.00619699934768428	Polynomial

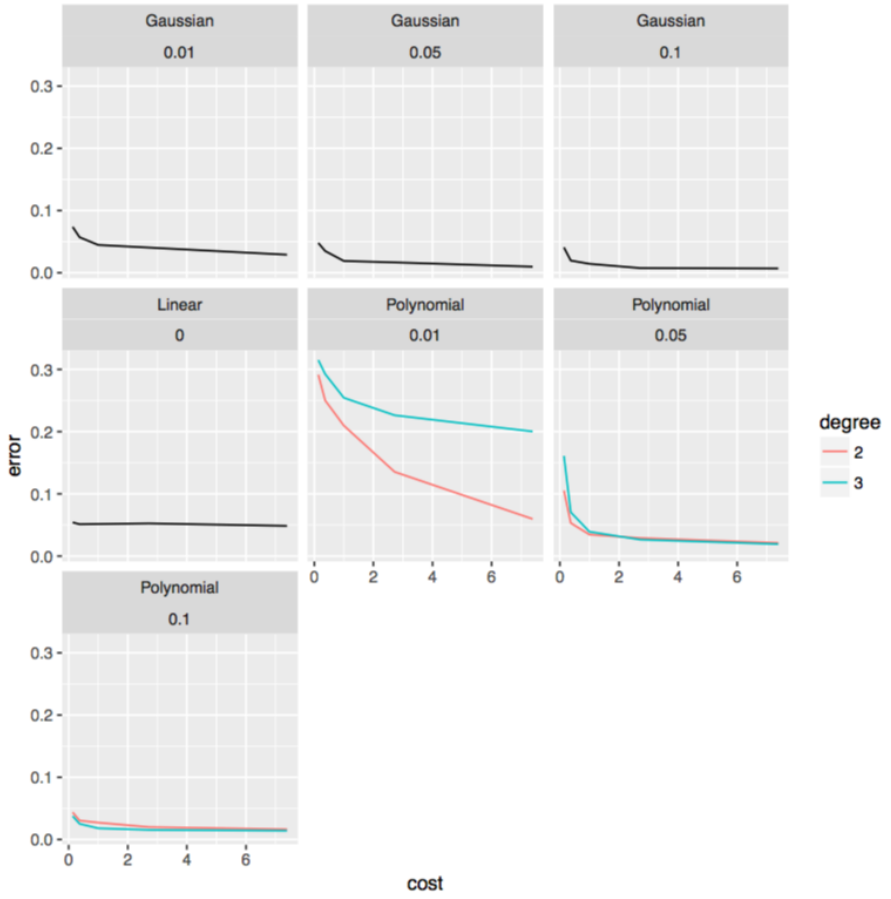
1-9 Ratio Cross-Validated SVM plots



Train/Test Best Error 1-9 Ratio

	Test Error	Train Error	Type
1	0.170143415906128	0.0156555772994129	Gaussian
2	0.170143415906128	0.0316373124592303	Linear
3	0.145371577574967	0.00424005218525766	Polynomial

Resampled Cross-Validated SVM plots



Train/Test Best Error Resampled

	Test Error	Train Error	Type
1	0.155149934810952	0	Gaussian
2	0.0827900912646675	0.0443574690150033	Linear
3	0.100391134289439	0	Polynomial

3 NEURAL NETWORKS

For our neural networks we examine how sensitive our results are to the number of hidden layers (both number of layers and nodes). We examine having between 1-2 hidden layers, combined with using 5, 10, and 15 nodes. This is done for our balanced model and all of our ratios and resampled models. Based on our optimal cross-validated model we compute the train and test errors.

Interestingly across all of my cross-validated neural networks, the 15 nodes, and 1 layer produced the optimal model, with error rates between 1.2 and 4.4 percent. The resampled data did the best for cross-validation error, followed by the 1-9, 2-8, 3-7, and finally balanced model. For testing error we found between 4.9 and 10.6 percent. While our optimal cross-validated model with the lowest error was the resampled, for our test error the balanced dataset has the lowest test error at 4.9 percent, followed by the 3-7, resampled, 1-9, and finally 2-8. This indicates some amount of overfitting in the training data particularly with unbalanced data. Finally we see that the 15 nodes being optimal and 1 layer show that the structure of the neural net is not particularly sensitive to the ratio of spam to not-spam data.

Normal Cross-Validated Neural Network Errors

	nnode	ntimes	avg error
1	15	1	0.0455562513320412
2	10	1	0.0483768931691291
3	15	2	0.0558979907281163
4	10	2	0.0581712667691027
5	5	1	0.0593706420659982
6	5	2	0.0670366715720005

Train/Test Best Error Balanced

	Error	Type
1	0.0136941636778611	Train Error
2	0.0495436766623207	Test Error

3-7 Ratio Cross-Validated Neural Network Errors

	nnode	ntimes	avg error
1	15	1	0.0318989925233363
2	5	1	0.0328548586269549
3	10	1	0.0328987830110154
4	15	2	0.0423804616772905
5	10	2	0.0443493761587315
6	5	2	0.0609477115831987

Train/Test Best Error 3-7 Ratio

	Error	Type
1	0.00782778864970646	Train Error
2	0.0749674054758801	Test Error

2-8 Ratio Cross-Validated Neural Network Errors

	nnode	ntimes	avg error
1	15	1	0.0301492774384514
2	10	1	0.0340843583760499
3	10	2	0.036601022121383
4	15	2	0.0375397410145596
5	5	1	0.0390128615079964
6	5	2	0.0486848374820473

Train/Test Best Error 2-8 Ratio

	Error	Type
1	0.00880626223091976	Train Error
2	0.10625814863103	Test Error

1-9 Ratio Cross-Validated Neural Network Errors

	nnode	ntimes	avg error
1	15	1	0.0220293102696324
2	10	1	0.0223906940379184
3	15	2	0.0249588865277241
4	5	1	0.0255964727006231
5	10	2	0.0315922797317861
6	5	2	0.036380929221619

Train/Test Best Error 1-9 Ratio

	Error	Type
1	0.00424005218525766	Train Error
2	0.0997392438070404	Test Error

Resampled Cross-Validated Neural Network Errors

	nnode	ntimes	avg error
1	15	1	0.0123052895627837
2	10	1	0.0175229155671879
3	5	1	0.0207500853844673
4	15	2	0.0252782566739327
5	10	2	0.0280787514482446
6	5	2	0.0364786733101962

Train/Test Best Error Resampled

	Error	Type
1	0.00684931506849315	Train Error
2	0.0853976531942634	Test Error

4 DECISION TREES

For our decision trees we examine how sensitive our results are to tree size. We examine between 0-6 maximum depth, with no restrictions on tree size being 0. This is done for our balanced model and all of our ratios and resampled models. Based on our optimal cross-validated model we compute the train and test errors.

Overall, we see the tree size sensitivity is consistent amount our different samples, as the shapes of our graphs (showing all of the cross-validated error for each of the 10 runs) are similar, with just varying levels of error.

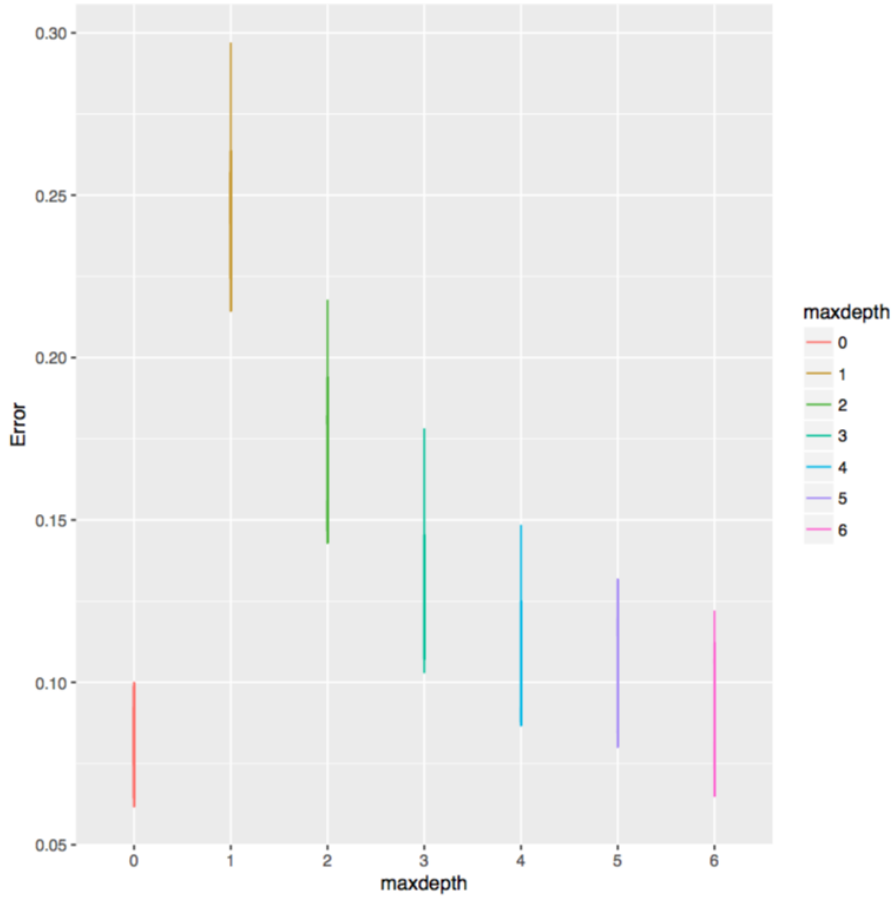
We always see that no restrictions on our model produce the best cross-validated error, although one then immediately suspects overfitting the training data. All of our graphs also show adding some kind of tree size restrictions creates much higher error, but as the restrictions increase, the range of errors drops steadily.

Interestingly, as we examine the range of our potential errors, we can see there are sometimes tree size restrictions that have less variability in cross validated errors than others.

Finally, we see based on our testing errors for our balanced model is about 46 percent, the 3-7 is around 15 percent, the 2-8 ratio around 20 percent, the 1-9 also around 20 percent, and the resampled with about 35 percent error.

Clearly the decision trees and having to split among so many different variables is suboptimal in terms of classification compared to SVM and neural networks, both of which produce fairly similar testing errors.

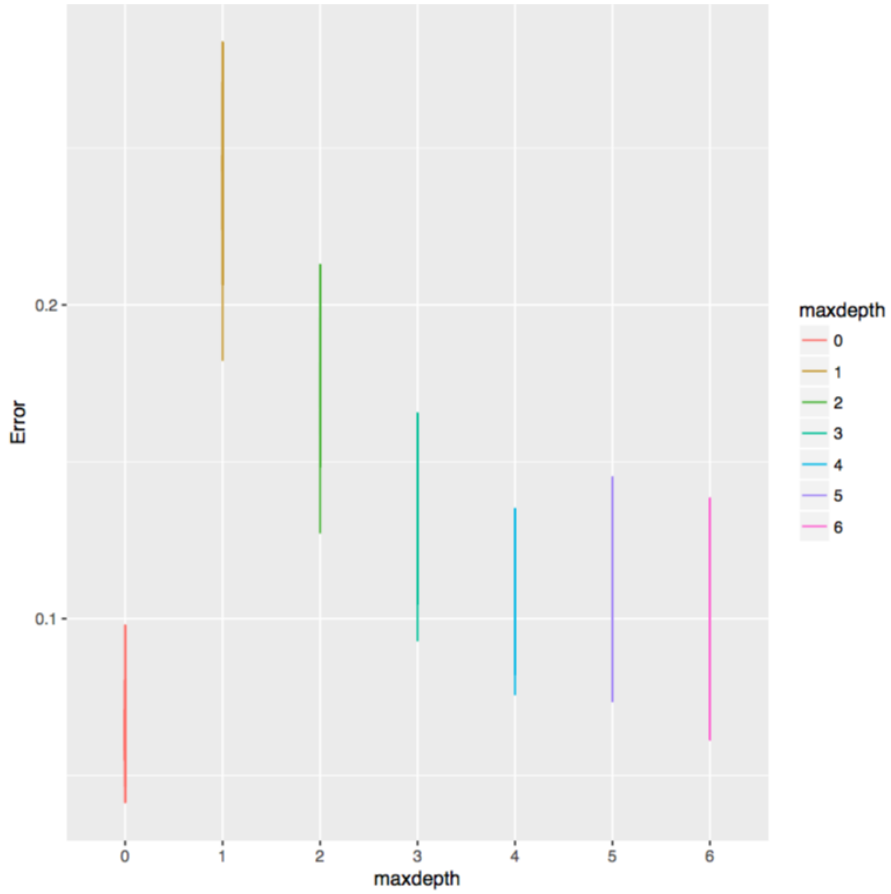
Balanced Cross-Validated Decision Tree plots



Train/Test Best Error Balanced

	Error	Type
1	0.0834691881317248	Train Error
2	0.467405475880052	Test Error

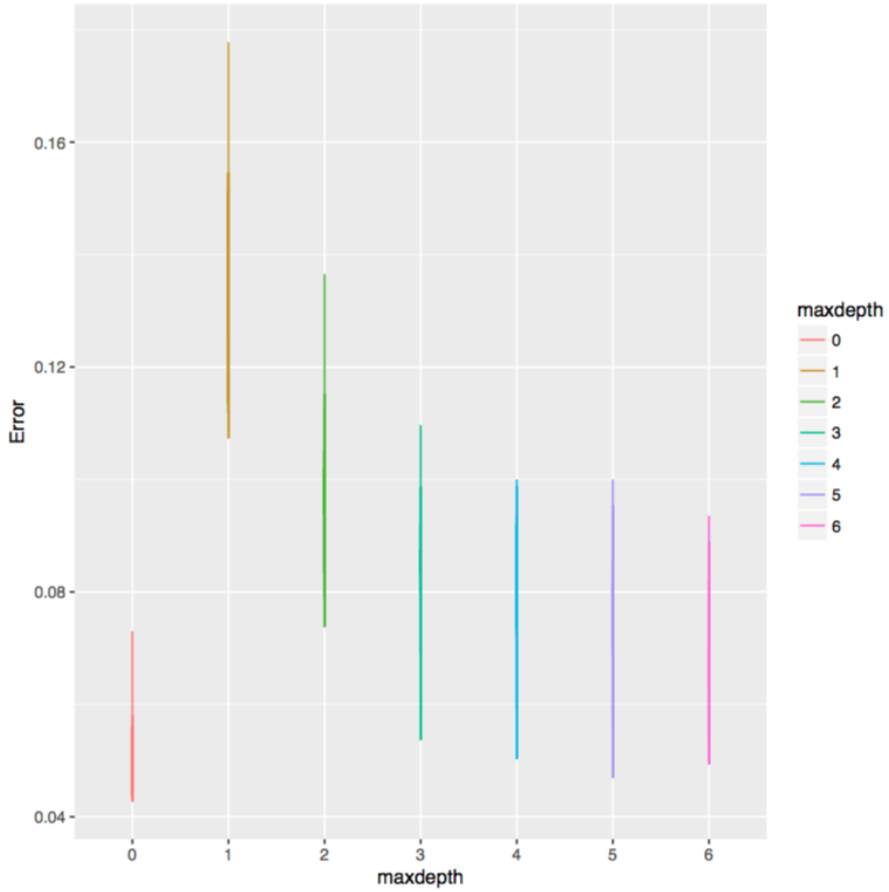
3-7 Ratio Cross-Validated Decision Tree plots



Train/Test Best Error 3-7 Ratio

	Error	Type
1	0.0639269406392694	Train Error
2	0.157757496740548	Test Error

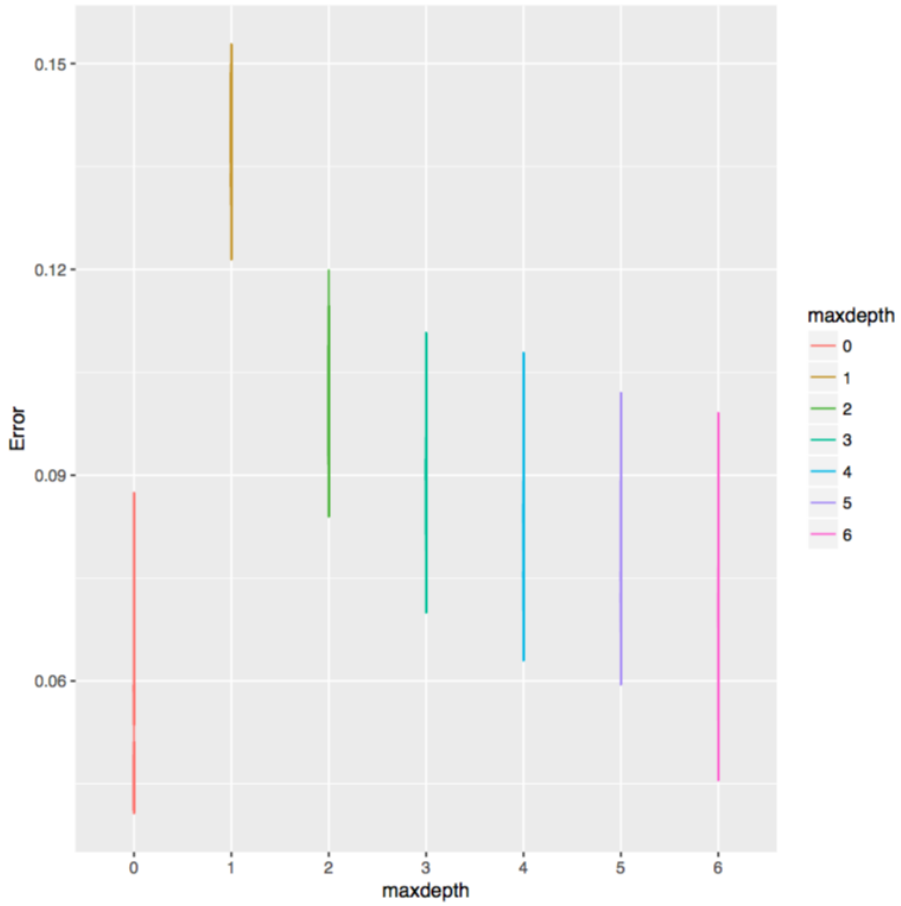
2-8 Ratio Cross-Validated Decision Tree plots



Train/Test Best Error 2-8 Ratio

	Error	Type
1	0.0534898891063275	Train Error
2	0.192959582790091	Test Error

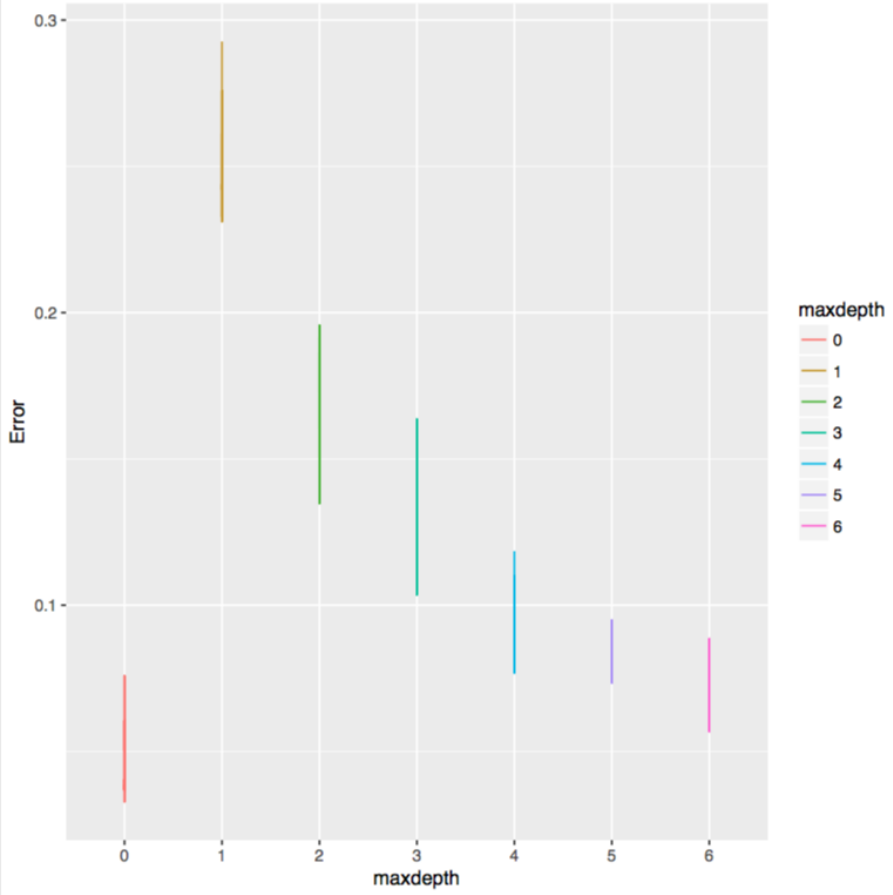
1-9 Ratio Cross-Validated Decision Tree plots



Train/Test Best Error 1-9 Ratio

	Error	Type
1	0.063449889348165	Train Error
2	0.204379582790091	Test Error

Resampled Cross-Validated Decision Tree plots



Train/Test Best Error Resampled

	Error	Type
1	0.0469667318982387	Train Error
2	0.353976531942634	Test Error