

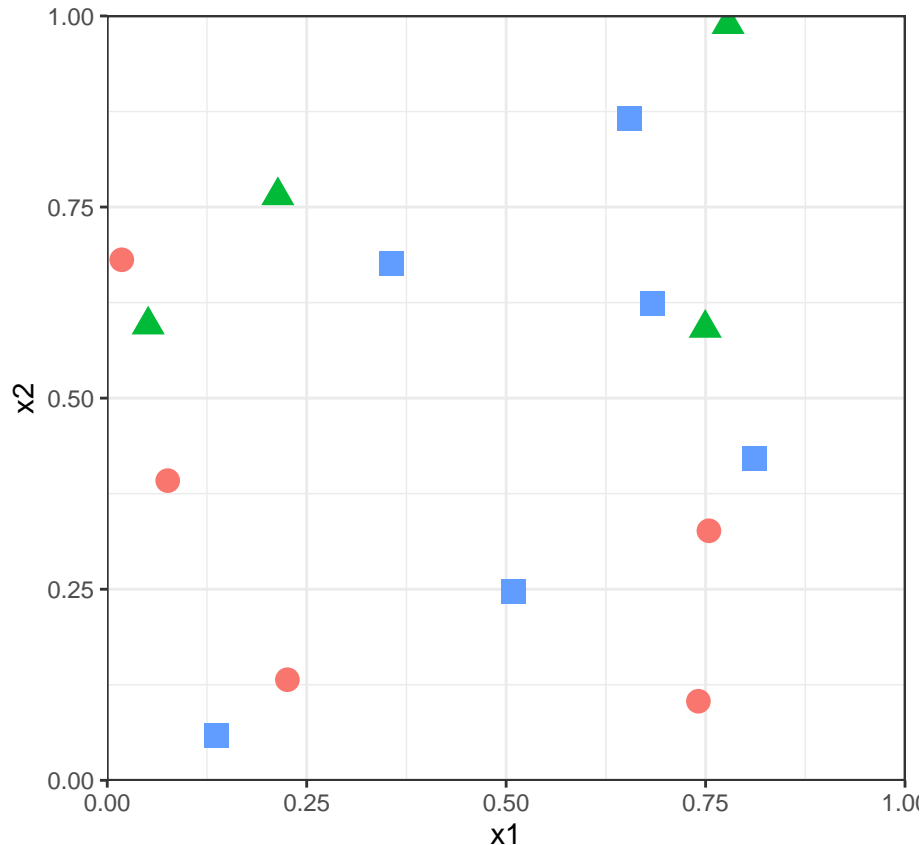
Lab 7

Alyssa Andrichik

11/13/2019

Lab 7: When a guest arrives they will count how many sides it has on

In class, we estimated by eye the first split in a classification tree for the following shapely data set. Now let's check to see if our graphical intuition agrees with that of the full classification tree algorithm.

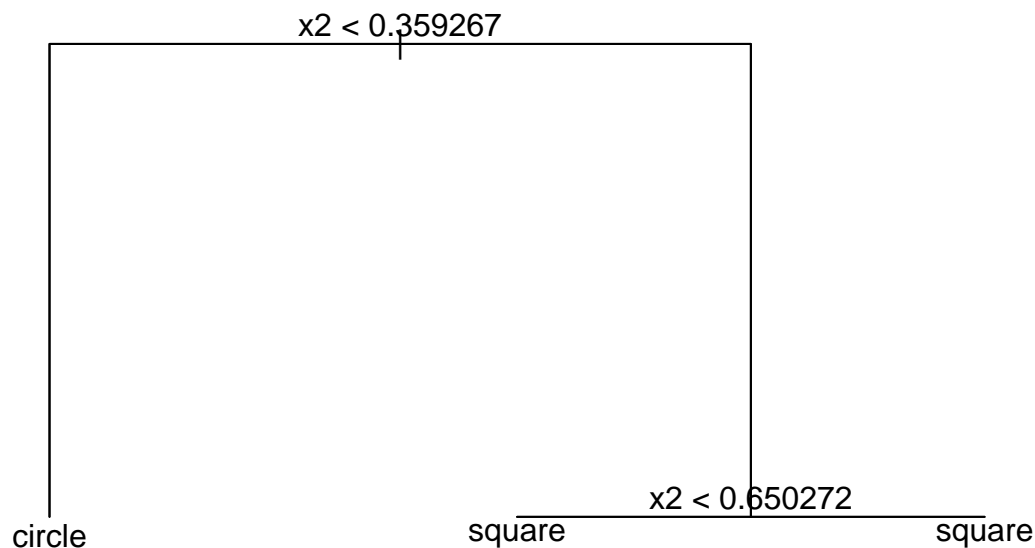


1. Growing the full classification tree

Use the `tree` package in R to fit a full unpruned tree to this data set, making splits based on the *Gini index*. You can find the code to do this in the slides from week 8 or in the lab at the end of Chapter 8 in the book. Please plot the resulting tree.

```
## -- Attaching packages ----- tidyverse 1.2.1 --  
  
## v tibble  2.1.1      v purrr   0.3.2  
## v tidyr   0.8.3      v dplyr  0.8.0.1  
## v readr   1.3.1      v stringr 1.4.0  
## v tibble  2.1.1      v forcats 0.4.0  
  
## -- Conflicts ----- tidyverse_conflicts() --  
## x dplyr::filter() masks stats::filter()
```

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



a. The two most common splits that we saw in class were a horizontal split around $X_2 \approx 0.50$ and a vertical split around $X_1 \approx 0.30$. Was either of these the first split decided upon by your classification tree?

No, the first split is $x_2 < 0.359267$

b. What is the benefit of the second split in the tree?

There is no benefit, if x_2 is larger than or less than 0.650272 it will be classified as a square.

c. Which class would this model predict for the new observation with $X_1 = 0.21, X_2 = 0.56$?

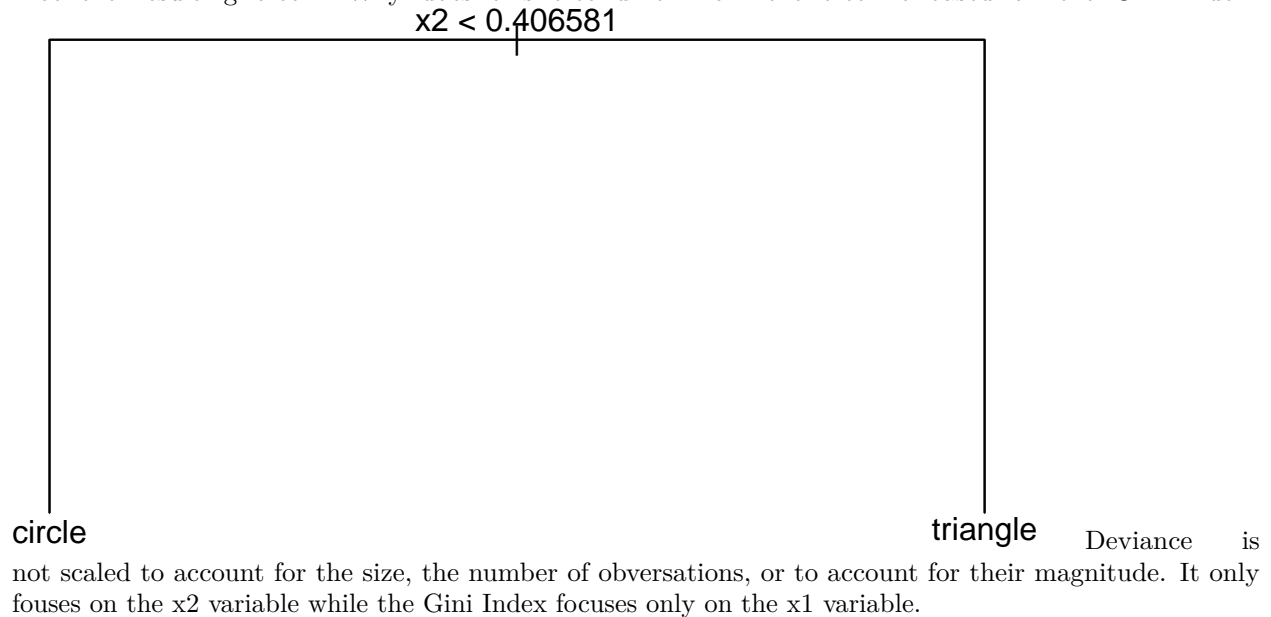
Square

2. An alternate metric

Now refit the tree based on the *deviance* as the splitting criterion (you set this as an argument to the `tree()` function). The deviance is defined for the classification setting as:

$$-2 \sum_m \sum_k n_{mk} \log \hat{p}_{mk}$$

Plot the resulting tree. Why does this tree differ from the tree fit based on the Gini Index?



Crime and Communities, revisited

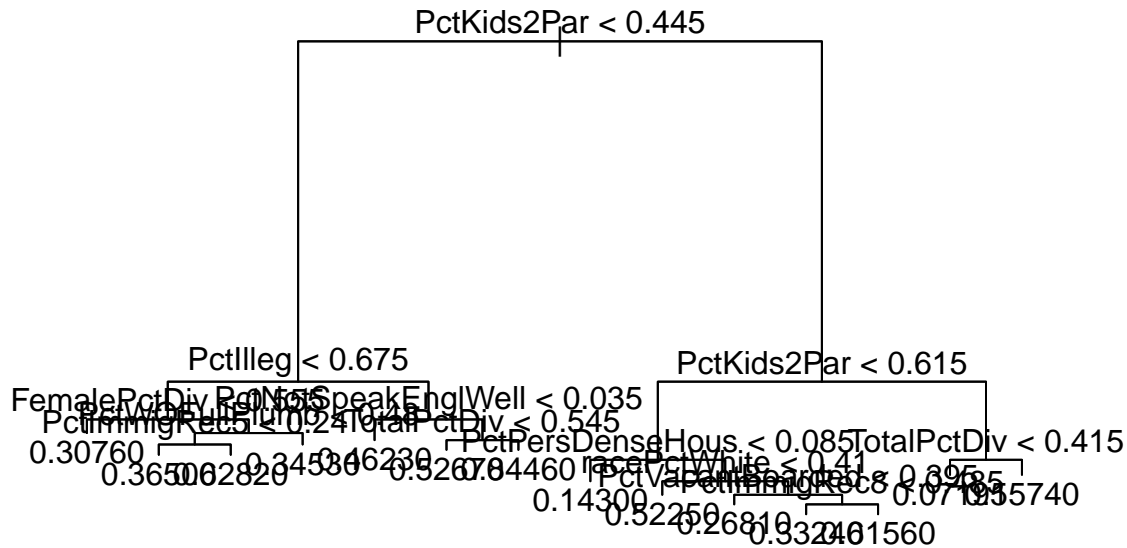
In Lab 3, you fit a regression model to a training data set that predicted the crime rate in a community as a function of properties of that community.

3. Growing a pruned regression tree

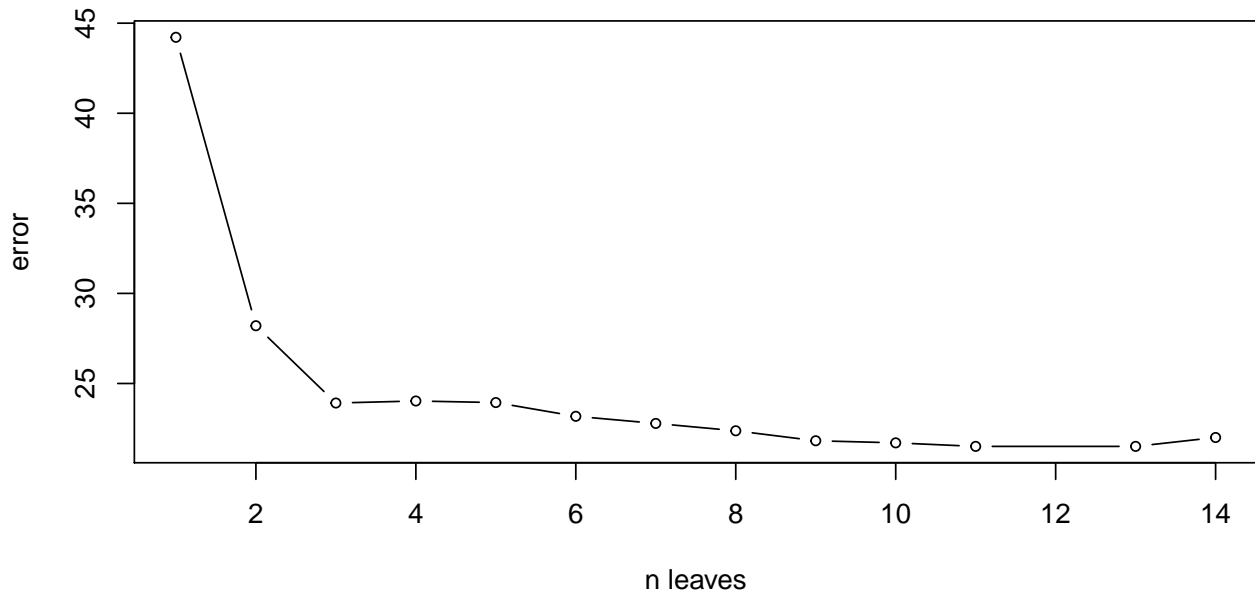
Fit a regression tree to the *training* data using the default splitting criteria (here, the deviance is essentially the RSS). Next, perform cost-complexity pruning and generate a plot showing the relationship between tree size and deviance to demonstrate the size of the best tree. Finally, construct the tree diagram for this best tree.

```
d <- read.csv("http://andrewpbray.github.io/data/crime-train.csv")
d[d=="?"] <- NA
d[d==""] <- NA

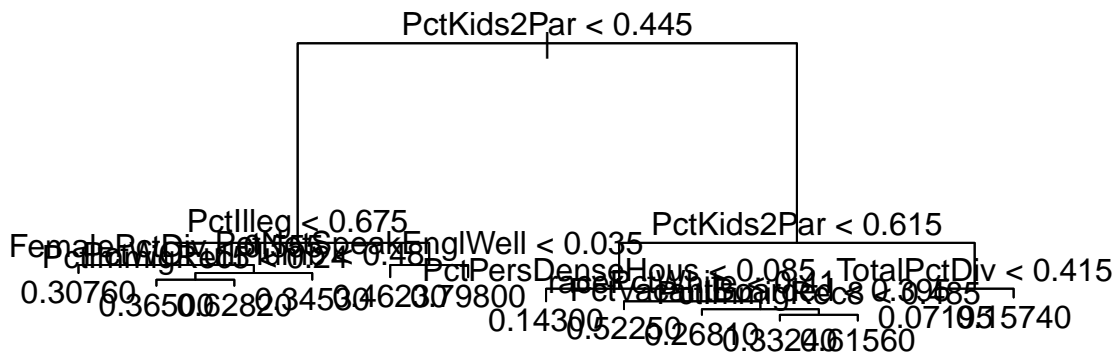
newd <- d %>%
  select(-c(state, county, community, communityname, population, LemasSwornFT, LemasSwFTPerPop,
    LemasSwFTFieldOps, LemasSwFTFieldPerPop, LemasTotalReq, LemasTotReqPerPop,
    PolicReqPerOffic, PolicPerPop, RacialMatchCommPol, PctPolicWhite, PctPolicBlack,
    PctPolicHisp, PctPolicAsian, PctPolicMinor, OfficAssgnDrugUnits, NumKindsDrugsSeiz,
    PolicAveOTWorked, PolicCars, PolicOperBudg, LemasPctPolicOnPatr, LemasGangUnitDeploy,
    PolicBudgPerPop)) %>%
  drop_na()
```



```
## $size
## [1] 14 13 11 10 9 8 7 6 5 4 3 2 1
##
## $dev
## [1] 22.00180 21.50829 21.50829 21.70486 21.82146 22.37553 22.78308
## [8] 23.17809 23.94086 24.02890 23.91528 28.20549 44.21623
##
## $k
## [1] -Inf 0.5176420 0.5222261 0.5544045 0.6118008 0.7251080
## [7] 0.7326129 0.8050502 1.1126087 1.1522867 2.0273988 4.1941945
## [13] 18.2672161
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune" "tree.sequence"
```



```
## [1] 13
```



4. Comparing predictive performance

Use this tree to compute the MSE for the *test* data set. How does it compare to the test MSE for your regression model? You can load the test data with the following code:

```
## [1] 0.01609171
```

##	2	7	10	12	13
##	0.5077756362	0.1615570852	0.1795327069	0.1596473765	0.1855321942
##	14	15	16	17	18
##	0.0977707260	0.0274269259	0.6140393420	0.0887156717	0.2413011778
##	24	25	26	27	28
##	0.2756914781	0.1804850461	0.0587666895	0.3333177318	0.0282717789
##	29	30	32	33	34
##	0.2387053299	0.6108551879	0.1228782809	0.1732975482	0.0367282902
##	36	38	39	40	41
##	0.1366395822	0.2705810767	0.1341163549	0.1084365643	0.1750330502
##	42	44	45	46	48
##	0.1121042630	0.0358421543	0.0025507056	0.0172657330	0.4786145554
##	50	51	52	53	54
##	0.4772403125	0.0866558071	0.2297997380	0.1293838005	0.1845492107
##	55	57	58	62	63

##	0.1201799772	0.1701217761	0.1941118604	0.0354456884	0.2298283019
##	65	74	76	77	78
##	0.2359131286	0.3739280393	0.2079520839	0.6050382894	0.6588015862
##	80	82	84	85	86
##	0.0173888880	0.1989682632	0.3999387859	0.3116705919	0.4604082926
##	88	96	98	99	102
##	0.2454571595	0.3280903009	0.0882523091	0.2871232473	0.1008405751
##	104	107	108	110	111
##	0.7883636290	0.0697171706	0.2699255819	0.6317559347	0.1483289933
##	113	114	115	117	119
##	0.4137805724	0.0590393069	0.1761895470	0.0676602559	0.2809039335
##	120	122	125	129	131
##	0.1165236106	0.1016847347	0.4346332175	0.1788772121	0.3858764794
##	134	135	136	137	138
##	0.2266523536	0.0515514972	0.3759736220	0.0350911988	0.0706445894
##	139	140	142	143	144
##	0.1778835901	0.1893636372	0.5833829436	0.3030262854	0.3348421305
##	145	146	147	148	151
##	0.1187322442	0.3314756132	0.0347915805	0.7145265131	0.2898927846
##	153	155	156	158	159
##	0.1392460686	0.2047890307	0.0221484432	0.1026534363	0.3833652775
##	160	161	166	167	168
##	0.4228242947	0.5125871128	0.2817345047	0.1977135319	0.3567303739
##	173	175	176	178	179
##	0.2525485033	0.5590789590	0.2769064896	0.1657031220	0.4410951978
##	183	184	185	187	188
##	0.6616139693	0.0863568822	0.3519159474	0.1827499380	0.5090545946
##	192	193	197	198	199
##	0.3043893724	0.0418686424	0.0940767200	0.8537705412	0.1169869732
##	200	203	204	207	209
##	0.2261499647	0.3952746913	0.1531411634	0.1501552669	0.0794143074
##	210	211	212	215	216
##	0.2934230463	0.0146080187	0.0998042833	0.4223046739	0.1123911624
##	218	219	221	223	226
##	0.1546821006	0.5103777857	0.2021832378	0.3137027622	0.0699634806
##	228	230	232	233	235
##	0.5005454685	0.1747033050	0.3244759105	0.0351467636	0.3987260308
##	237	240	242	245	249
##	0.3356899336	0.5145231290	0.2507363356	0.4123748156	0.0683939795
##	251	253	255	256	258
##	0.0320896335	0.3533366796	0.0841325798	0.0802185711	0.2432351136
##	259	260	267	268	269
##	0.3295373405	0.4518422148	0.6445251770	0.4377300674	0.3630301729
##	270	271	275	277	283
##	0.1369534824	0.0675904092	-0.0016216385	-0.0057819572	0.1941089104
##	289	290	292	293	295
##	0.2947598259	0.1375527189	0.3932831104	0.1487082273	0.7135705304
##	297	299	302	303	305
##	0.2300032021	0.0348321699	0.0966525621	0.3883321164	0.2277029274
##	306	308	310	314	315
##	0.3375973859	0.2904237374	0.2126027661	0.1308564541	0.1538522229
##	316	326	331	334	342
##	0.3761786492	0.4393671588	0.1514371752	0.0443249729	0.2270181752
##	343	344	346	347	348

##	0.0261863003	0.1451487082	0.3330961661	0.1550230017	-0.0399039770
##	350	352	355	358	364
##	0.1281424816	0.2630505973	0.1991447265	0.1739539126	0.0771793664
##	365	366	370	371	373
##	0.2633502156	0.6601660601	0.2269775859	0.3699763912	0.0930937365
##	376	377	378	379	380
##	0.0664309624	0.2811194231	0.1544380471	0.4465118108	0.2143487305
##	382	383	386	387	390
##	0.2700194796	0.5254216888	0.4775662381	0.7367720199	0.0398913434
##	392	393	394	395	396
##	0.1032955187	-0.0057000850	0.2290923218	0.0937748452	0.2009988705
##	398	399	404	405	408
##	0.1451636836	-0.0329350947	0.6933216349	0.2688593393	0.6840998862
##	413	418	419	424	426
##	0.0989301728	0.0600080084	0.3247365025	0.1996749859	0.2476785120
##	431	434	437	439	440
##	0.1559647024	0.3581916955	0.1048094550	0.0684368253	0.1606559739
##	441	444	445	448	449
##	0.1284443563	0.0392102347	0.0427960611	0.3264291586	0.2573080585
##	451	454	455	460	465
##	0.1590601654	0.1509062224	0.2546390123	0.1543824823	0.1085860267
##	468	469	470	471	473
##	0.0289408622	0.6337346207	-0.0545514142	0.4005209666	0.2501754319
##	474	475	476	477	480
##	0.2887573887	0.5143616411	0.3299743957	0.1431863846	0.1440319312
##	481	482	483	489	490
##	0.0650664885	0.0911833342	0.1084095635	0.0114150147	0.3107318412
##	491	492	493	494	497
##	0.2093308398	0.0279578787	0.1699017735	0.3495308505	0.1222917632
##	498	502	503	504	505
##	0.0995053585	0.1491452825	0.0890978556	0.3699051576	0.3510597623
##	507	509	515	516	517
##	0.4074245152	0.0782149647	0.5749940226	0.2120981206	0.3081260482
##	519	523	525	527	529
##	0.4002970950	0.2128640515	0.4125948182	0.3511116837	0.2373971141
##	530	533	534	535	536
##	0.0951551642	-0.0006003222	0.1769817853	0.7838773847	0.0708652854
##	537	545	547	548	549
##	0.4942833089	0.1468270824	0.2497134562	0.5597893251	0.3705157261
##	550	554	555	557	558
##	0.7376581558	0.1049446355	0.2311912128	0.8623495139	0.1198504081
##	561	562	563	564	565
##	0.0436821970	0.0663078074	0.4259986799	0.1699573382	0.2948011087
##	566	567	568	570	571
##	0.4190567985	0.1771169657	0.2368008276	0.0705506917	0.0816093525
##	572	573	575	576	577
##	0.0935307917	0.4498821477	0.4016157731	0.1294799547	-0.0144968039
##	578	581	582	587	589
##	0.4480790555	0.2768254870	-0.0041185584	0.1959367470	0.3175625930
##	590	591	593	595	597
##	0.2783784497	0.3484426374	0.3223064793	0.1704882911	0.0523324035
##	599	602	605	606	611
##	0.1915593758	0.2549778330	0.3382095175	0.1439223647	0.4340031605
##	612	613	615	616	622

```
## 0.2191908514 0.2304831032 0.5083578170 0.0892060352 0.4008244044
##      623      627      631      632      633
## 0.2400591653 0.5109402525 0.3452346580 0.3062306214 0.0629540090
##      635      637      638      639      640
## 0.2625715658 0.3330172439 0.3152713938 0.1961567497 0.2271427171
##      642      643      650      651      653
## 0.1566067848 0.2811194231 0.1127876282 0.3022739430 0.0990832787
##      655      657      658      661      662
## 0.2443133816 0.0868472457 0.1212824723 0.3675313927 0.1546430743
##      663      664      667      668      671
## 0.0146479146 0.1834723295 0.3249685305 0.2133416961 0.1702862139
##      674      675      676      677      678
## 0.0602806258 0.2746318289 0.2095778433 0.3996241922 0.4708157954
##      679      681      686      687      688
## 0.4173942693 0.2380918114 0.3012773710 0.1699573382 0.1489275364
##      691      695      699      700      701
## 0.0984135019 0.1617065476 0.1658931737 0.0566249526 0.3474972933
##      702      703      705      710      714
## 0.1418482181 0.0399874975 0.3804507909 -0.0228970569 0.4338679801
##      715      717      718      719      720
## 0.1238619579 0.0309451621 0.0541835975 0.1573847411 0.4576117544
##      721      725      727      728      730
## 0.2586169677 0.0970347460 0.7132738621 0.3131582208 0.0665804248
##      731      733      736      739      740
## 0.1719759201 0.2394547223 0.2315613713 0.7662426641 0.3131845283
##      741      744      745      746      747
## 0.1640930314 0.2039149202 0.2340839051 0.1305425538 0.1272706276
##      749      753      756      757      759
## 0.4715810328 0.2654363876 0.0219982874 0.2570916994 0.2326225835
##      760      761      763      765      767
## 0.2895382950 0.3622529101 0.8151203007 0.2228871139 0.0006139959
##      775      778      783      784      785
## 0.2481711320 0.4137655970 0.2314082654 0.3191057868 0.1985987982
##      786      788      791      792      794
## 0.0535024888 0.0644672519 0.1976443786 0.3533772690 0.0519885524
##      797      798      800
## 0.4366389043 0.4633542931 0.2264466330
## [1] 0.02320641
```

The MSE for my tree model is 0.01609171 and the MSE for my linear regression model is 0.02320641. The predictive performance of the tree model is better than the regression model since the mean squared error is smaller.

5. Growing a random forest

We now apply methods to decrease the variance of our estimates. Fit a `randomForest()` model that performs only bagging and no actual random forests (recall that bagging is the special case of random forests with $m = p$). Next, fit a second random forest model that uses $m = p/3$. Compute their test MSEs. Is this an improvement over the vanilla pruned regression tree? Does it beat your regression model?

```
## [1] 12 26 9 47 16 41 96 81 43 79 29 65 13 36 38 54 73
## [18] 18 3 24 63 92 25 27 15 80 21 83 76 40 97 62 32 35
## [35] 94 14 61 85 48 49 78 88 60 31 50 28 52 34 53 37 74
## [52] 82 98 91 6 39 56 7 57 33 19 87 64 30 66 59 1 46
```



```
## [69] 42 71 69 86 75 93 58 100 90 77 20 10 84 55 17 99 11
## [86] 45 4 44 23 67 70 22 95 2 72 5 8 51 89 68
```

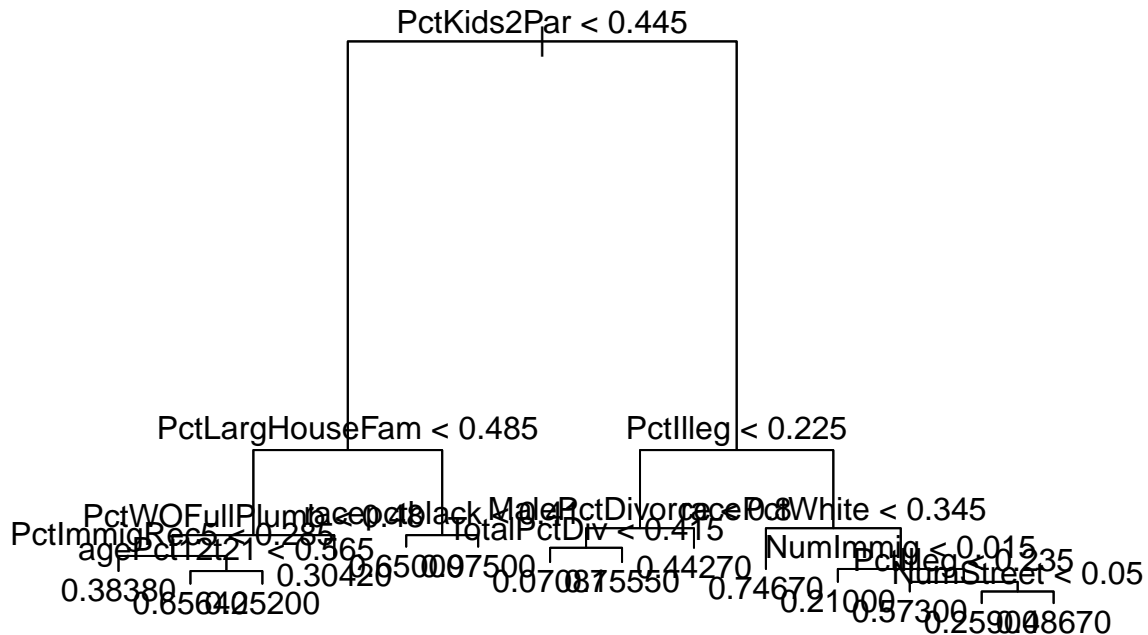
```
#First Random Split
```

```
library(tree)
```

```
rftree <- tree(ViolentCrimesPerPop ~ .-ViolentCrimesPerPop,
               data = crim_rforest)
```

```
plot(rftree)
```

```
text(rftree, pretty = 0)
```



```
##
```

```
## Call:
```

```
## randomForest(formula = ViolentCrimesPerPop ~ . - ViolentCrimesPerPop, data = crim_rforest, imp
```

```
## Type of random forest: regression
```

```
## Number of trees: 500
```

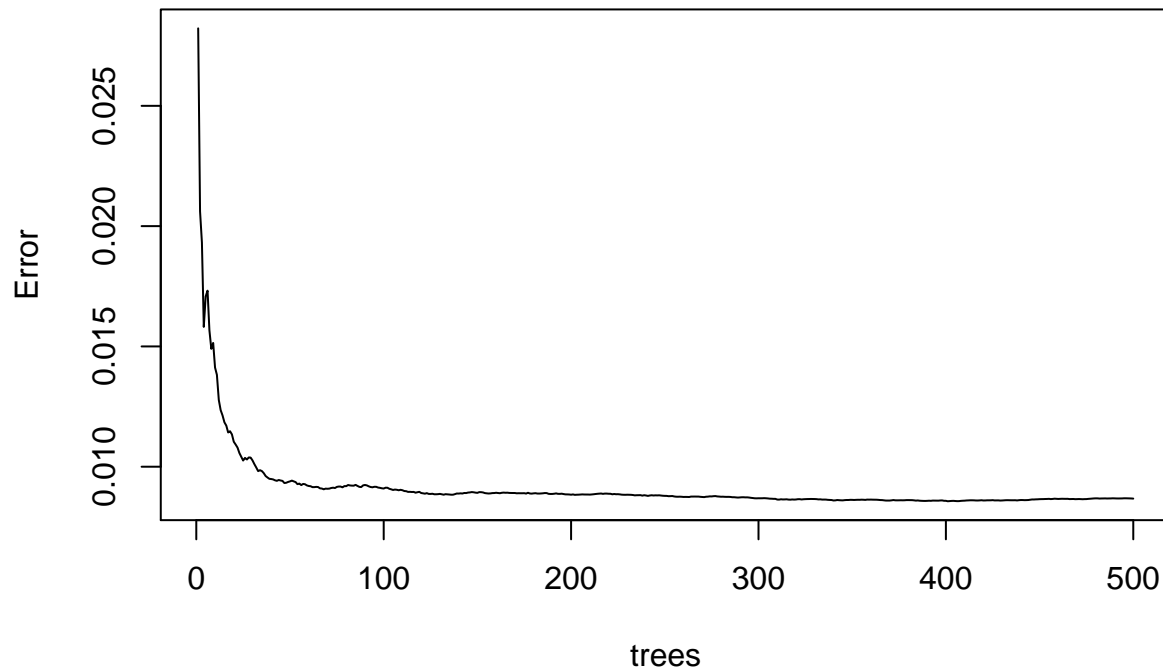
```
## No. of variables tried at each split: 33
```

```
##
```

```
## Mean of squared residuals: 0.008674571
```

```
## % Var explained: 86.98
```

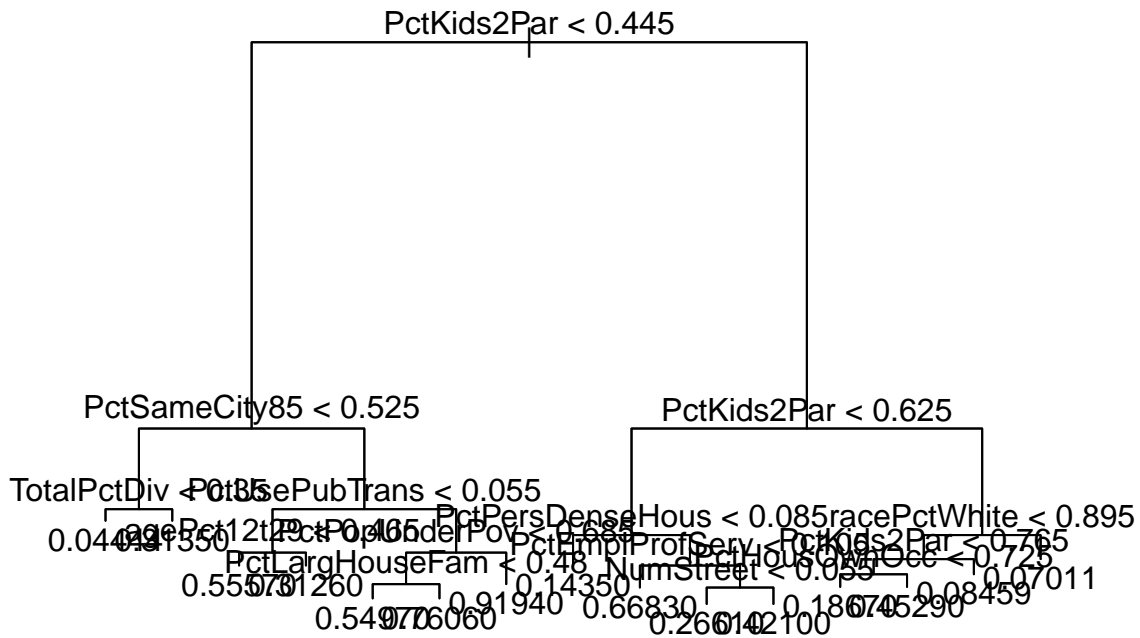
model1



```
## [1] 0.001341236
```

```
##      [1] 87 42 82 53 31 40 5 45 97 37 77 6 62 69 21 30 65
##     [18] 90 55 20 86 32 14 11 26 34 50 72 60 49 36 70 47 19
##    [35] 17 66 98 85 73 80 41 16 27 43 100 29 81 2 88 74 22
##    [52] 23 38 18 59 12 54 25 52 99 39 71 4 68 57 92 9 89
##    [69] 67 96 56 44 28 75 10 3 64 35 63 24 51 48 33 79 58
##   [86] 7 83 46 13 84 8 15 93 1 76 78 91 61 94 95
```

```
#First Random Split
rftree2 <- tree(ViolentCrimesPerPop ~ .-ViolentCrimesPerPop,
               data = crim_rforest2)
plot(rftree2)
text(rftree2, pretty = 0)
```

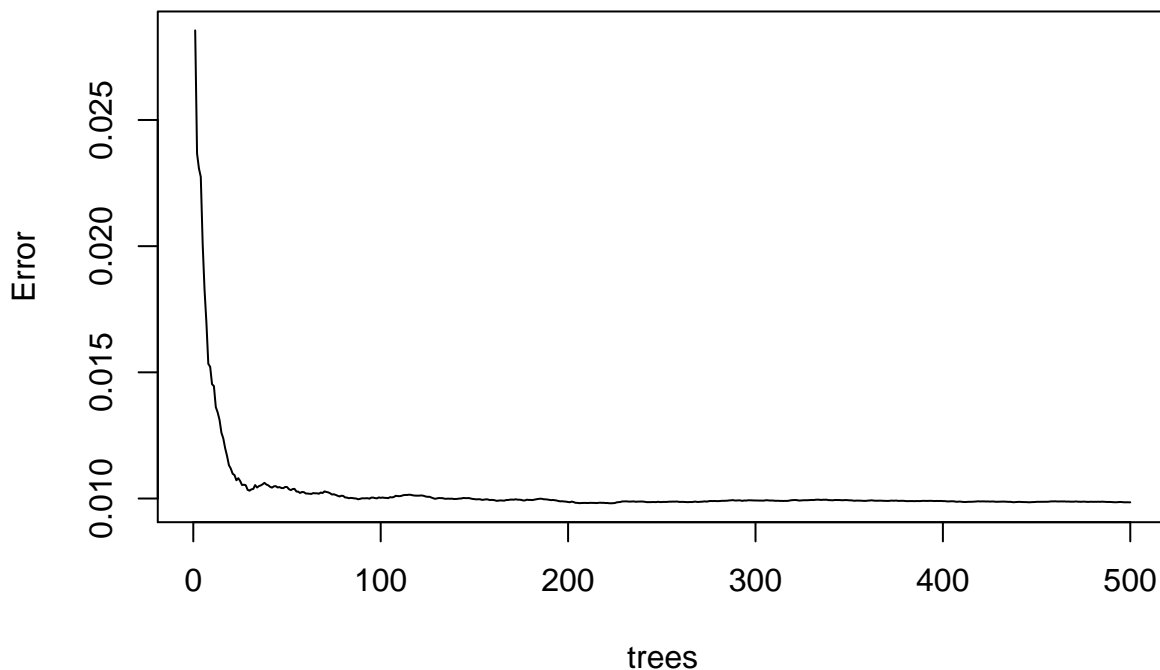


```

##
## Call:
## randomForest(formula = ViolentCrimesPerPop ~ . - ViolentCrimesPerPop,      data = crim_rforest2, imp
##           Type of random forest: regression
##           Number of trees: 500
## No. of variables tried at each split: 33
##
##           Mean of squared residuals: 0.009848266
##           % Var explained: 80.07

```

model2



```
## [1] 0.1073736
```

The test MSE for the random forest with $m = p$ is 0.001511666. The test MSE for the random forest with $m = p/3$ is 0.001485386. Yes, the test MSE's from these random forests are much smaller than the singular pruned regression tree and my regression model, thus the predictive performance of the random forests is higher than all other methods.

6. Variance importance

One thing we lose by using these computational techniques to limit the variance is the clearly interpretable tree diagram. We can still salvage some interpretability by considering `importance()`. Please construct a Variable Importance Plot (`varImpPlot()`). Are these results similar/different from your interpretation of your regression coefficients in Lab 3?

##	%IncMSE	IncNodePurity
## medIncome	6.858889	0.09081510
## HispPerCap	7.986216	0.21745097
## agePct65up	6.408117	0.10351589
## PctWorkMomYoungKids	8.610607	0.22077344
## pctWSocSec	6.969395	0.08977911
## TotalPctDiv	13.109434	0.75754594
## LandArea	6.929442	0.14807714
## OwnOccHiQuart	5.321689	0.06389573
## PctFam2Par	13.464068	4.24204188
## OwnOccLowQuart	4.671621	0.05975020
## PctLess9thGrade	6.896907	0.15924047
## PersPerOwnOccHous	6.576430	0.13575957
## pctWWage	5.422339	0.08757720
## PctOccupManu	7.507206	0.17830847
## MalePctDivorce	14.834560	0.63753429
## PctImmigRec8	8.333166	0.25111644
## PctHousOwnOcc	9.849114	0.16562367
## pctWRetire	7.396598	0.14504682
## racePctWhite	14.923650	3.03763705
## AsianPerCap	9.344940	0.24754819
## PctLargHouseOccup	10.094698	0.66123513
## PctBornSameState	7.244834	0.13321280
## OtherPerCap	9.364202	0.21137791
## NumUnderPov	9.137963	0.36128725
## pctWInvInc	7.531950	0.70621797
## OwnOccMedVal	5.001567	0.06818987
## whitePerCap	5.220883	0.10001072
## RentMedian	3.912584	0.07589785
## MedYrHousBuilt	8.555089	0.11296170
## FemalePctDiv	12.188727	0.54393630
## PopDens	6.440771	0.12792815
## PctLargHouseFam	14.851255	1.57273738
## PctUnemployed	8.273687	0.14961831
## PctEmplProfServ	7.665463	0.12880425
## PctSameCity85	7.329234	0.13930508
## pctWFarmSelf	8.872675	0.20577278
## PctNotSpeakEnglWell	9.853215	0.40340010
## MedRent	5.300422	0.08243621
## PctWorkMom	9.969609	0.18672818
## NumIlleg	10.212157	1.28483882

## PctWOFullPlumb	10.556328	0.26383559
## MedOwnCostPctIncNoMtg	8.754623	0.22777281
## PctSpeakEnglOnly	9.568818	0.26276351
## PctBSorMore	9.269272	0.20625479
## PctIlleg	20.589666	8.50646416
## PctPopUnderPov	5.118246	0.37358288
## PctImmigRecent	8.120799	0.19538349
## PctEmplManu	9.892605	0.25186262
## PctImmigRec5	9.218459	0.27734310
## PctOccupMgmtProf	8.460065	0.27997620
## PctVacantBoarded	9.490273	0.33495296
## RentLowQ	5.885813	0.08694492
## PctUsePubTrans	9.104368	0.21435058
## PctForeignBorn	8.339506	0.25043842
## agePct12t21	6.097533	0.11164408
## MalePctNevMarr	6.263827	0.24675743
## PctRecentImmig	7.477419	0.24248424
## agePct12t29	8.457887	0.15558959
## PctRecImmig5	6.188038	0.14169678
## PctEmploy	6.274396	0.08298581
## medFamInc	6.043858	0.10817869
## MedOwnCostPctInc	8.300451	0.18832516
## PersPerOccupHous	6.350249	0.10480105
## PctNotHSGrad	5.393053	0.24996089
## PersPerRentOccHous	6.867739	0.15012659
## PctRecImmig10	6.914944	0.13206538
## householdsize	8.077629	0.12472826
## PctTeen2Par	8.658936	1.11988995
## PersPerFam	7.698397	0.32424964
## HousVacant	9.827179	0.32051894
## PctHousLess3BR	10.330663	0.37166656
## MedRentPctHousInc	9.354407	0.24480423
## PctVacMore6Mos	7.387387	0.09898758
## PctSameHouse85	8.410314	0.12990667
## PctRecImmig8	6.080823	0.13963158
## NumStreet	11.966957	0.50637704
## PctHousNoPhone	7.563916	0.15654896
## perCapInc	5.915022	0.07061432
## numbUrban	8.269738	0.19298424
## RentHighQ	4.875286	0.08348235
## PctImmigRec10	7.618937	0.19507328
## pctWPubAsst	5.782141	0.43448420
## LemasPctOfficDrugUn	6.273897	0.07518718
## pctUrban	4.137235	0.05045731
## PctYoungKids2Par	9.581060	2.24365851
## racePctAsian	8.267959	0.12767041
## PctKids2Par	21.514627	9.86820639
## indianPerCap	8.641653	0.19043901
## PctPersOwnOccup	8.552604	0.27069970
## MedNumBR	2.669066	0.02687316
## blackPerCap	7.899180	0.15597152
## PctSameState85	8.928798	0.15774152
## racepctblack	12.821883	1.40950633
## PctHousOccup	11.100250	0.26006245

model1

