**Prediction Lab**

1. Redfin Data
   1. Identify your most parsimonious model from the Categorical Variables Lab

The parsimonious model from the categorical lab has the following variables:

lm(formula = PriceinThousand ~ Baths + Beds + Sqft + LOTSIZE + YearBuilt + Redfin\_lab3$`PropertyTypeCondo/Co-op` + Redfin\_lab3$`PropertyTypeSingle Family Residential` + Redfin\_lab3$PropertyTypeTownhouse + Redfin\_lab3$`LocationBelvedere Terrace` + Redfin\_lab3$LocationBryant + Redfin\_lab3$`LocationGreen Lake` + Redfin\_lab3$`LocationHawthorne Hills` + Redfin\_lab3$LocationInverness + Redfin\_lab3$`LocationMaple Leaf` + Redfin\_lab3$`LocationMatthews Beach` + Redfin\_lab3$LocationMeadowbrook + Redfin\_lab3$LocationNorthgate + Redfin\_lab3$LocationRavenna + Redfin\_lab3$LocationRoosevelt + Redfin\_lab3$`LocationSand Point` + Redfin\_lab3$LocationSeattle + Redfin\_lab3$`LocationView Ridge` + Redfin\_lab3$LocationWedgwood + Redfin\_lab3$LocationWindermere, data = Redfin\_lab3)

So, the variables are Baths, Beds, Square Feet, Lot Size, Year Built, Property Type, Location Category.

* 1. Predict the housing price, manually, for 3 different sets of values (i.e. 4 beds, 3 baths, 2400 sq. foot, 6000 lot, single family home, built in 2012, in Alki neighborhood).

baths<-redfin$Baths

beds<-redfin$Beds

sqft<-redfin$Sqft

lotsize<-redfin$LOTSIZE

year<-redfin$YearBuilt

property<-redfin$`PROPERTY TYPE`

location<-redfin$LOCATION

testprice<-lm (redfin$PriceinThousand ~ baths+beds+sqft+lotsize+year+property+location)

new.value<-data.frame(baths = c(1,2,4) , beds= c(2,4,6) , sqft= c(1000,1800,2400), lotsize= c (4500, 6000, 9000), property= c("Condo/Co-op", "Single Family Residential", "Single Family Residential"), location= c("Bryant", "Green Lake", "Seattle"), year= c(1973,1998, 2000))

predict (testprice, newdata = new.value, se.fit = TRUE)

I did predict for the following criteria:

* 1 bath, 2 beds, 1000 Sq. feet, 4500 lot, Condo Property type, built in 1973 in Bryant Neighborhood. ***It did predict a price of $404,437.7***
* 2 baths, 4 beds, 1800 Sq. feet, 6000 lot, Single Family Property type, built in 1998 in Green Lake Neighborhood. ***It did predict a price of $998,217.6***
* 4 baths, 6 beds, 2400 Sq. feet, 9000 lot, Single Family Property type, built in 2000 in Seattle Neighborhood. ***It did predict a price of $1,101,647.***

$fit

1 2 3

404.4377 998.2176 1101.6473

$se.fit

1 2 3

47.47438 50.53033 95.37784

$df

[1] 253

$residual.scale

[1] 148.867

* 1. Use inverse prediction to predict the number of beds, baths, and square foot for the prices you got in part b.

invBed<-lm(redfin$PriceinThousand ~beds)

inverse.predict(invBed,404.4377)

$Prediction

[1] 0.761267

$`Standard Error`

[1] 1.535338

$Confidence

[1] 3.946712

$`Confidence Limits`

[1] -3.185446 4.707979

For 2 beds, it does give an inverse prediction of 0.76 beds, which is not good prediction thus the results are not satisfactory for this model.

inverse.predict(invBed,998.2176)

$Prediction

[1] 3.970531

$`Standard Error`

[1] 1.461872

$Confidence

[1] 3.757862

$`Confidence Limits`

[1] 0.2126686 7.7283934

For 4 beds, it does give an inverse prediction of 3.97 beds, which is good prediction and thus the results are satisfactory for this model.

inverse.predict(invBed,1102.6473)

$Prediction

[1] 4.532837

$`Standard Error`

[1] 1.495061

$Confidence

[1] 3.843177

$`Confidence Limits`

[1] 0.6896592 8.3760142

For 6 beds, it does give an inverse prediction of 4.5 beds, which is not a good prediction and thus the results are not satisfactory for this model.

Baths

invBath<-lm(redfin$PriceinThousand ~baths)

inverse.predict(invBath,404.4377)

$Prediction

[1] 0.8134355

$`Standard Error`

[1] 0.730766

$Confidence

[1] 1.567337

$`Confidence Limits`

[1] -1.053902 2.080773

For 1 bath, it does give an inverse prediction of 0.8 bath, which is a good prediction and thus the results are satisfactory for this model.

inverse.predict(invBath,998.2176)

$Prediction

[1] 2.133377

$`Standard Error`

[1] 0.6629973

$Confidence

[1] 1.421988

$`Confidence Limits`

[1] 1.191389 4.035365

For 2 baths, it does give an inverse prediction of 2.13 baths, which is a good prediction and thus the results are satisfactory for this model.

inverse.predict(invBath,1102.6473)

$Prediction

[1] 3.981044

$`Standard Error`

[1] 0.6616049

$Confidence

[1] 1.419001

$`Confidence Limits`

[1] 1.562042 4.400045

For 4 baths, it does give an inverse prediction of 3.98 baths, which is a good prediction and thus the results are satisfactory for this model.

So overall, baths have quite good prediction.

Square Feet

invSQFT<-lm(redfin$PriceinThousand ~sqft)

inverse.predict(invSQFT,404.4377)

$Prediction

[1] 653.621

$`Standard Error`

[1] 545.4637

$Confidence

[1] 1074.375

$`Confidence Limits`

[1] -420.7539 1727.9959

For 1000 Sq.Ft, it does give an inverse prediction of 653.6 Sq.Ft, which isn’t good prediction thus the results are not satisfactory for this model.

inverse.predict(invSQFT,998.2176)

$Prediction

[1] 2461.107

$`Standard Error`

[1] 543.3487

$Confidence

[1] 1070.209

$`Confidence Limits`

[1] 1390.898 3531.316

For 1800 Sq.Ft, it does give an inverse prediction of 2461 Sq.Ft, which isn’t good prediction and thus the results are not satisfactory for this model.

inverse.predict(invSQFT,1102.6473)

$Prediction

[1] 2777.569

$`Standard Error`

[1] 543.8543

$Confidence

[1] 1071.205

$`Confidence Limits`

[1] 1706.365 3848.774

For 2400 Sq.Ft, it does give an inverse prediction of 2777.5 Sq.Ft, which isn’t good prediction thus the results are not satisfactory for this model.

* 1. Use the predict function to predict the price for a different zip code of data

(download diff zip code and predict)

Downloaded a new dataset from redfin for Bellevue and included all the variables i.e. baths, beds, sqft, lotsize, year, Property type as all were significant variable and were even applicable in the new data set as well. I removed location from my new data. This is because the names of the neighborhoods are not the same in the new zip code.

I did make the property types as a factor.

redfinnew$PropertyTypes <- as.factor(redfinnew$`PROPERTY TYPE`)

pricefit<-lm(redfin$PriceinThousand ~ baths+beds+sqft+lotsize+year+redfin$PropertyTypes)

baths1<-redfinnew$BATHS

beds1<-redfinnew$BEDS

sqft1<-redfinnew$`SQUARE FEET`

lotsize1<-redfinnew$`LOT SIZE`

year1<-redfinnew$`YEAR BUILT`

property1<-redfinnew$PropertyTypes

predict.data<-data.frame(redfinnew$PRICE, baths1, beds1, sqft1, lotsize1, year1, property1)

predict\_new\_zip<-predict (pricefit, predict.data)

predict\_new\_zip

Output

|  |  |
| --- | --- |
| Original Dataset Values  Price in Thousand | Predicted Values  Price in thousand |
| 275 | 601 |
| 770 | 478 |
| 603 | 757 |
| 461 | 704 |
| 450 | 683 |
| 400 | 503 |
| 489 | 784 |
| 420 | 950 |
| 516 | 758 |
| 496 | 643 |
| 500 | 701 |
| 742 | 537 |
| 830 | 548 |
| 500 | 879 |
| 655 | 820 |
| 889 | 896 |
| 665 | 769 |
| 465 | 1058 |
| 777 | 1137 |
| 525 | 836 |
| 350 | 408 |
| 855 | 790 |

The above prediction of price in thousand for the new data model is not accurate as the predicted prices in thousand are not same as the ones in the dataset and that can be inferred from the table above. So, the parsimonious model of the previous model doesn’t do a good job of predicting prices for the new dataset.

1. Seattle Rain Data
   1. Using the model, you found in the Logistic Regression Lab, predict the presence (yes or no) of rain on 12/1/2017, using values from a weather site for the variables you included in the final model.

The parsimonious model found from the Logistic Regression Lab has following variables:

**SeasonCategory, Ave Wind, TMAX, TMIN, WSF5**

rainseason<-RainSeattle2016$SeasonCategories

raintempmax<-RainSeattle2016$TMAX

raintempmin<-RainSeattle2016$TMIN

rainavewind<-RainSeattle2016$`Ave Wind`

gust<-RainSeattle2016$WSF5

rainlogit<-glm(RainSeattle2016$Rain~rainseason+ raintempmax+ raintempmin+ rainavewind+ gust, family=binomial)

predict.rain<-data.frame(rainseason="Fall", raintempmax=48, raintempmin=41, rainavewind=10 ,gust=22)

predict (rainlogit,predict.rain, type="response")

1

0.7898517

It did rain on 1st of December 2017 so the model is a decent one in predicting the rain with a probability of 0.79. Thus it is a good fit model.

f. Using the model of rain in the Time Series model, forecast the rain amount for 12/1/2017 (the 335th day of the year)

As, the data for the rain dataset was until 31st December 2016, we would consider predicting 355 ahead as we need to get the rain amount for 12/1/2017.

prcpFore<- predict(prcpmodel, 335)

prcpFore

$pred

Time Series:

Start = 1.00049603174603

End = 1.91306433775696

Frequency = 366

[1] 0.16167129 0.17406039 0.13446067 0.11990243 0.07823764 0.18125347 0.18268380 0.05172474

[9] 0.12812510 0.18996177 0.11514447 0.11726123 0.12432420 0.13969906 0.16655958 0.09934893

[17] 0.10775711 0.17808380 0.12929480 0.10510241 0.14142332 0.13660685 0.14087860 0.12665951

[25] 0.11156876 0.15373263 0.14309161 0.10699092 0.13650834 0.14414633 0.12806848 0.13152739

[33] 0.12495969 0.13671937 0.14406750 0.11862494 0.12809464 0.14563587 0.12868430 0.12752470

[41] 0.13362969 0.13203754 0.13763073 0.12870817 0.12582569 0.14058482 0.13336435 0.12543787

[49] 0.13465773 0.13379704 0.13219082 0.13245475 0.12857724 0.13489862 0.13551502 0.12730061

[57] 0.13230099 0.13551376 0.13080062 0.13192947 0.13177426 0.13220437 0.13456989 0.13034057

[65] 0.13069144 0.13505960 0.13183729 0.13077407 0.13300081 0.13214910 0.13280746 0.13213588

[73] 0.13084191 0.13353263 0.13289615 0.13069384 0.13264710 0.13283027 0.13189394 0.13240128

[81] 0.13174726 0.13240823 0.13303491 0.13140100 0.13201392 0.13305174 0.13193413 0.13204600

[89] 0.13236146 0.13211578 0.13260958 0.13206702 0.13181428 0.13276232 0.13228096 0.13183676

[97] 0.13243894 0.13227452 0.13222714 0.13231234 0.13198872 0.13238888 0.13245409 0.13192931

[105] 0.13226672 0.13242783 0.13212518 0.13225932 0.13221144 0.13220889 0.13240016 0.13212666

[113] 0.13214217 0.13241794 0.13219972 0.13216157 0.13230033 0.13221218 0.13227879 0.13224758

[121] 0.13214492 0.13232076 0.13227974 0.13214357 0.13227561 0.13226814 0.13221318 0.13226312

[129] 0.13220628 0.13224492 0.13229416 0.13218701 0.13222857 0.13229048 0.13221539 0.13223435

[137] 0.13225003 0.13222572 0.13226604 0.13223179 0.13221189 0.13227393 0.13224078 0.13221605

[145] 0.13225605 0.13223857 0.13223904 0.13224842 0.13222288 0.13224902 0.13225439 0.13222030

[153] 0.13224363 0.13225120 0.13223146 0.13224398 0.13223845 0.13223668 0.13225144 0.13223329

[161] 0.13223414 0.13225181 0.13223674 0.13223619 0.13224495 0.13223719 0.13224306 0.13224160

[169] 0.13223381 0.13224555 0.13224271 0.13223419 0.13224327 0.13224155 0.13223832 0.13224260

[177] 0.13223797 0.13224035 0.13224404 0.13223688 0.13223981 0.13224352 0.13223840 0.13224038

[185] 0.13224110 0.13223903 0.13224218 0.13223993 0.13223846 0.13224259 0.13224025 0.13223889

[193] 0.13224158 0.13224001 0.13224027 0.13224109 0.13223915 0.13224090 0.13224130 0.13223905

[201] 0.13224069 0.13224100 0.13223970 0.13224076 0.13224024 0.13224004 0.13224115 0.13223991

[209] 0.13223998 0.13224112 0.13224007 0.13224016 0.13224071 0.13224008 0.13224057 0.13224049

[217] 0.13223992 0.13224072 0.13224051 0.13223997 0.13224060 0.13224041 0.13224022 0.13224056

[225] 0.13224020 0.13224036 0.13224062 0.13224014 0.13224035 0.13224057 0.13224022 0.13224040

[233] 0.13224043 0.13224026 0.13224050 0.13224035 0.13224024 0.13224052 0.13224035 0.13224028

[241] 0.13224046 0.13224033 0.13224037 0.13224043 0.13224029 0.13224041 0.13224043 0.13224028

[249] 0.13224040 0.13224041 0.13224032 0.13224041 0.13224036 0.13224035 0.13224043 0.13224034

[257] 0.13224035 0.13224042 0.13224035 0.13224036 0.13224040 0.13224035 0.13224039 0.13224038

[265] 0.13224034 0.13224040 0.13224038 0.13224035 0.13224039 0.13224037 0.13224036 0.13224039

[273] 0.13224036 0.13224037 0.13224039 0.13224036 0.13224037 0.13224039 0.13224036 0.13224038

[281] 0.13224038 0.13224036 0.13224038 0.13224037 0.13224036 0.13224038 0.13224037 0.13224037

[289] 0.13224038 0.13224037 0.13224037 0.13224038 0.13224037 0.13224038 0.13224038 0.13224037

[297] 0.13224038 0.13224038 0.13224037 0.13224038 0.13224037 0.13224037 0.13224038 0.13224037

[305] 0.13224037 0.13224038 0.13224037 0.13224037 0.13224038 0.13224037 0.13224037 0.13224037

[313] 0.13224037 0.13224038 0.13224037 0.13224037 0.13224038 0.13224037 0.13224037 0.13224037

[321] 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037

[329] 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037 0.13224037

$se

Time Series:

Start = 1.00049603174603

End = 1.91306433775696

Frequency = 366

[1] 0.2220032 0.2283495 0.2340989 0.2343173 0.2346999 0.2348917 0.2362849 0.2396796 0.2397178

[10] 0.2398409 0.2441021 0.2445277 0.2446019 0.2456689 0.2464865 0.2479050 0.2482759 0.2483629

[19] 0.2505861 0.2516407 0.2516840 0.2526798 0.2538071 0.2544090 0.2550940 0.2554321 0.2564446

[28] 0.2576991 0.2578908 0.2584944 0.2597936 0.2603068 0.2608641 0.2615378 0.2622092 0.2631820

[37] 0.2636598 0.2641107 0.2651992 0.2658590 0.2663011 0.2670693 0.2677580 0.2684537 0.2691057

[46] 0.2696032 0.2704078 0.2711744 0.2716357 0.2723138 0.2730724 0.2736668 0.2743142 0.2749160

[55] 0.2755689 0.2763109 0.2768586 0.2774510 0.2782008 0.2788068 0.2793977 0.2800519 0.2806724

[64] 0.2813359 0.2819489 0.2825209 0.2832056 0.2838435 0.2844098 0.2850506 0.2856832 0.2862928

[73] 0.2869161 0.2875056 0.2881319 0.2887728 0.2893481 0.2899550 0.2905908 0.2911846 0.2917875

[82] 0.2923924 0.2929914 0.2936101 0.2942003 0.2947858 0.2954050 0.2960002 0.2965836 0.2971855

[91] 0.2977779 0.2983722 0.2989649 0.2995444 0.3001409 0.3007347 0.3013095 0.3018974 0.3024867

[100] 0.3030660 0.3036495 0.3042278 0.3048072 0.3053919 0.3059641 0.3065380 0.3071195 0.3076916

[109] 0.3082629 0.3088367 0.3094064 0.3099786 0.3105471 0.3111116 0.3116819 0.3122485 0.3128102

[118] 0.3133755 0.3139387 0.3144999 0.3150615 0.3156197 0.3161791 0.3167386 0.3172933 0.3178492

[127] 0.3184055 0.3189585 0.3195115 0.3200636 0.3206142 0.3211654 0.3217140 0.3222613 0.3228099

[136] 0.3233560 0.3239008 0.3244459 0.3249893 0.3255321 0.3260741 0.3266144 0.3271548 0.3276943

[145] 0.3282318 0.3287694 0.3293061 0.3298415 0.3303764 0.3309101 0.3314430 0.3319756 0.3325066

[154] 0.3330369 0.3335669 0.3340955 0.3346233 0.3351506 0.3356767 0.3362023 0.3367269 0.3372506

[163] 0.3377738 0.3382960 0.3388172 0.3393379 0.3398577 0.3403767 0.3408950 0.3414123 0.3419289

[172] 0.3424449 0.3429599 0.3434742 0.3439878 0.3445005 0.3450126 0.3455238 0.3460343 0.3465441

[181] 0.3470531 0.3475613 0.3480689 0.3485756 0.3490816 0.3495870 0.3500915 0.3505954 0.3510985

[190] 0.3516009 0.3521026 0.3526036 0.3531038 0.3536034 0.3541023 0.3546004 0.3550979 0.3555946

[199] 0.3560907 0.3565861 0.3570808 0.3575748 0.3580681 0.3585607 0.3590527 0.3595440 0.3600346

[208] 0.3605246 0.3610139 0.3615025 0.3619905 0.3624778 0.3629644 0.3634504 0.3639358 0.3644205

[217] 0.3649046 0.3653880 0.3658708 0.3663529 0.3668344 0.3673153 0.3677956 0.3682752 0.3687542

[226] 0.3692326 0.3697104 0.3701875 0.3706641 0.3711400 0.3716153 0.3720900 0.3725641 0.3730376

[235] 0.3735105 0.3739828 0.3744545 0.3749256 0.3753961 0.3758661 0.3763354 0.3768042 0.3772723

[244] 0.3777399 0.3782070 0.3786734 0.3791393 0.3796046 0.3800693 0.3805335 0.3809971 0.3814601

[253] 0.3819226 0.3823845 0.3828458 0.3833066 0.3837669 0.3842266 0.3846857 0.3851443 0.3856024

[262] 0.3860599 0.3865168 0.3869733 0.3874291 0.3878845 0.3883393 0.3887936 0.3892474 0.3897006

[271] 0.3901533 0.3906055 0.3910571 0.3915083 0.3919589 0.3924090 0.3928586 0.3933076 0.3937562

[280] 0.3942042 0.3946518 0.3950988 0.3955453 0.3959913 0.3964369 0.3968819 0.3973264 0.3977704

[289] 0.3982139 0.3986570 0.3990995 0.3995416 0.3999831 0.4004242 0.4008648 0.4013049 0.4017445

[298] 0.4021837 0.4026224 0.4030605 0.4034983 0.4039355 0.4043723 0.4048086 0.4052444 0.4056797

[307] 0.4061146 0.4065491 0.4069830 0.4074165 0.4078496 0.4082821 0.4087143 0.4091459 0.4095771

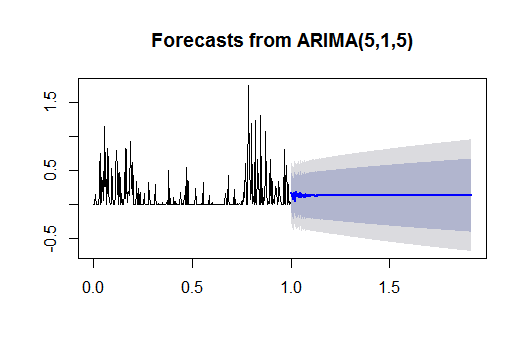
[316] 0.4100079 0.4104382 0.4108681 0.4112975 0.4117264 0.4121549 0.4125830 0.4130106 0.4134378

[325] 0.4138645 0.4142908 0.4147167 0.4151421 0.4155671 0.4159917 0.4164158 0.4168395 0.4172627

[334] 0.4176856 0.4181080

From the weather underground the precipitation on 12/1/2017 was 0.27, whereas the time series model predicts the precipitation value to be 0.13, thus this model doesn’t seem to be quite accurate in predicting.

plot(prcpFore)



mean(RainSeattle2016$PRCP)

[1] 0.1234426

The forecast has the prediction line along the mean value of the Rain data of 2016, but it doesn’t really give the range of values. The graph shows that the 80th percentile is well in the boundaries of confidence interval (-0.5 to 0.5). Values are being predicted in the beginning but later they coincide to predict a single value. The data on which this model is built is of just one year and therefore`, do not have seasonality. This could be the reason behind not getting the exact values on the later date.