Bayesian Project Report

Aaron Niecestro

December 6, 2019

There have been lots of models, reports, and ideas of how people think about what should determine the price of a house. This report is similar, but it is only focusing on the price of residential property in the District of Columbia. I have read lots of literature about pricing models not just about in DC but all over the United States and world, so I felt it was time to branch out this topic into a Bayesian model. Before making this Bayesian model I have made at least a hundred different models, some of which included linear, polynomial and logistic type, to figure out what variables can be used to determine a price of a property in DC. So now I am creating a Bayesian model to not just increase my knowledge about statistics, or also attempt to see what new and intriguing insights I can gather from making a pricing Bayesian model. So, my question is what variables determine the price of a property in DC? I am interested in this question because I live in DC and I once was thinking of moving to DC for work, but the cost of living in DC was way too high to afford regardless of the income I would earn at my new job.

The data I used in this Bayesian model and analysis was obtained from Kaggle[1]. After I downloaded the data, I started to assess what each column in the dataset was trying to represent. This was a rigorous task since one of the columns in the dataset was qualified, and I had no idea what exactly this column meant or was trying to explain. To fix this problem I tried to reach out to the people who put up the dataset on Kaggle and I got an answer a couple of months later. The Qualified column meant whether a property in DC was worthy of selling or not. However, it should be noted now that I do not know how this variable was exactly determined. So, I assume that it was determined by the DC housing guidelines and passing all the inspections. I have kept this assumption constant throughout the entire analysis. The dataset from Kaggle had originally 158957 rows and 49 columns, but I do not wish to look at all that data so I filtered and changed these so I could work with the dataset I felt best suited my needs. This smaller dataset I worked on was called the final dataset and I used this dataset for my analysis. This dataset had 57610 rows and 21 columns, but I did not stop there because I broke the dataset in half so I could create the training and testing datasets. I felt that creating a 50% training and 50% testing dataset was the best way to split the data at random because I believed that my predictions would be most accurate and trustworthy with only half of the data.

[1] https://www.kaggle.com/christophercorrea/dc-residential-properties

Since the dataset had 49 columns, this report will describe only the variables used in the models and analysis. The model used the following response and predictor variables:

1. Price_10K – the price of the most recent sale in 10 thousand dollars. The average price of a property in DC in ten thousand dollars was $57.725, and the range was from $1.027 to $910.000 (in $10,000).
2. Bathrooms – the number of full bathrooms the property has. The average number of bathrooms was 2.522, and the number of bathrooms ranged from 0 to 11.
3. Rooms, the number of rooms the property has. The average number of rooms was 7.438, and the number of bathrooms ranged from 0 to 30.
4. Bedrooms, the number of bedrooms the property has. The average number of bedrooms was 3.422, and the number of bathrooms ranged from 0 to 15.
5. Kitchens, the number of kitchens. The average number of kitchens was 1.247, and the number of kitchens ranged from 0 to 6.
6. Fireplaces, the number of fireplaces. The average number of fireplaces was 0.6422, and the number of fireplaces ranged from 0 to 9.
7. AYB.age, the number of years since the earliest time the main portion of the building was built. The average number of AYB.age was 84.58 years, and the number of AYB.age ranged from 1 to 254.00 years.
8. EYB.age, the number of years since the improvement was built more recent than actual year built. The average number of EYB.age was 49.24 years, and the number of EYB.age ranged from 1 to 86 years.
9. Remodel.age, the number of years since the property was last remodeled. The average number of remodeling .age was 8.398, and the number of remodeling.age ranged from 0 to 139.
10. Condition, a categorical rating of the condition of the property.
11. AC, a categorical rating of whether a property has AC or not.
12. Grade, a categorical rating of the grade of the property.

As I have mentioned earlier, there were more variables, but these variables described above were used in my analysis and model and should be explained. I do not believe that it is necessary to explain the variables I did not use. Although if one wishes to know about more of the variables, I have attached the link to Kaggle on the first footnote.

Unfortunately, when I was looking through the original dataset from the missing dataset there was a lot of missing data. The amount of missing data was 562000 observations in the original Kaggle dataset. One of the reasons the dataset had so many missing observations was because two of the 49 columns in the downloaded Kaggle dataset were empty. So, I took these two empty columns out before doing any type of analysis. This was so much missing data that if I choose only the variables I wish to use and attempted to case-wise delete on only these variables I have approximately 20% of the original dataset. This is 20% of the original dataset before I even after I was able to clean up the data. For Example, each column besides the ID column had missing rows between at least 1 observation to over nearly seventy percent missing. To fix and remedy this problem of so much missing data observation, I tried to substitute 0 into the missing data observations and filter the data again after this change was attempted. I know that this is not the best solution, it was one of the few remedies I could think of when I started my analysis.

Before beginning the analysis, I looked at the distributions of the variables and normality of the data. I found that most of the data and variables were not normally distributed. It was also interesting that the histograms showed me that most of my data is either very skewed to the left or the right (see appendix Graph #1).

I believed that the Bayesian model I created was an appropriate modeling strategy because I have attempted every other statistical model and analysis which made sense in figuring out the price of a property in DC. (See appendix table #4). The Bayesian model I created was

$$Y_i \sim \alpha_j + \beta_{j0[i]} + \beta_{j1[i]} + \beta_{j2[i]} + \beta_{j3[i]} + \beta_{j4[i]} + \beta_{j5[i]} + \beta_{j6[i]} + \beta_{j7[i]} + \beta_{j8[i]} + \beta_{j9[i]} + \beta_{j10[i]} + \beta_{j11[i]} + \beta_{j12[i]} + \beta_{j13[i]} + \beta_{j14[i]} + \beta_{j15[i]} \text{ for } j = 1, 2, \ldots , 8.$$

It seems that my model passed the MCMC chains. (see appendix Table #3). Although the model did pass the MCMC non-convergence tests, not all the betas were normally distributed. It seems that some of the betas were slightly skewed either to the right or left, but the beta distributions still followed a "normal" distribution.

The created Bayesian model was as follows:

Price_10K = - 0.920544 + 21.137957*Bathrooms + 0.575739*Bedrooms + 0.220931*Rooms - 4.707369*Kitchens + 14.905539*Fireplaces + 0.466959*AYB.age - 0.849885*EYB.age - 0.123803*Remodel.age - 5.168444*AC=Yes - 64.649561*Condition=Excellent + 0.851290*Condition=Fair - 1.928609*Condition=Good - 43.564921*Condition=Very Good + 6.387614*REMODEL.age*CONDITION=Excellent - 0.180788*REMODEL.age*CONDITION=Fair + 0.128282*REMODEL.age*CONDITION=Good + 1.319853*REMODEL.age*CONDITION=Very Good + 0.005704*ROOMS*AYB.age + 0.937902*ROOMS*AC=Yes + 28.054394*BATHRM*CONDITION=Excellent + 2.305573*BATHRM*CONDITION=Fair + 3.230037*BATHRM*CONDITION=Good + 18.855757*BATHRM*CONDITION=Very Good

The DIC of this model is 284692.4. I know there are smaller DIC models, but this was the smallest DIC model that did not overcomplicate the model by having too many variables. I did not wish to create a model that follows the R^2 principle, with the more you the better the R^2 one can get. I was afraid of multicollinearity between the variables. I know for a fact when making the previous model using this dataset that was similar models which had multicollinearity. The credible intervals show that the model betas are not good, cannot be fully trusted, and have room to improve. This is because a lot of the credible intervals the model outputted shows that many of the betas cross 0 and this is not something I wished to see. For example, the bedrooms and kitchens credible intervals had values that were across 0. (see appendix Table #2). I believe that although my model is satisfactory, it is not the best model, and needs to be improved. This is because the standard deviations for the beta are small, the standard deviation for the intercept is huge, and most of the variables are not statistically significant. Since most of the betas are not statistically significant, this worries me. I have more confidence in a model if a majority of the variables in the model were statistically significant and the standard deviations were reasonable (see appendix table #2). One of the few saving graces forms my model is that the beta chain distributions were very similar to one another and that there were no iterations that skewed the graphs (see appendix Graph #2).

In conclusion, it seems that a Bayesian model does help explain the price of a property in DC. By having the variables grouped by grade, I was able to see that and conclude that the variance best explained this way when used with a linear model. Although this model may not be the "best" model to explain the price of a property in DC, this model does get closer to which variables are important and which grouping is best to use. While the data and the model may not be normally distributed, and even the transformation for the variables could not get the distributions to be normally distributed. The bright side to all this is that from appendix table #4, this Bayesian model is one way to figure out how to explain the price of a property in DC, and more analysis will need to be completed before finalizing my analysis on model to predict the price of a property in the District of Columbia.

Appendix

Table #1 Bayesian Model

Random effects:

Groups	Name	Variance	Std. Dev.
BATHRM	(Intercept)	5798	76.14
Residual		1244	35.27

Fixed effects:

	Estimate	Std. Error	t value
(Intercept)	-0.920544	34.769284	-0.026
BATHRM	21.137957	5.549337	3.809
BEDRM	0.575739	0.277716	2.073
ROOMS	0.220931	0.355394	0.622
KITCHENS	-4.707369	0.472762	-9.957
FIREPLACES	14.905539	0.274742	54.253
AYB.age	0.466959	0.027037	17.271
EYB.age	-0.849885	0.022123	-38.417
REMODEL.age	-0.123803	0.022298	-5.552
ACY	-5.168444	1.674129	-3.087
CONDITIONExcellent	-64.649561	4.986097	-12.966
CONDITIONFair	0.851290	4.895098	0.174
CONDITIONGood	-1.928609	1.325022	-1.456
CONDITIONVery Good	-43.564921	2.453195	-17.758
REMODEL.age:CONDITIONExcellent	6.387614	0.457848	13.951
REMODEL.age:CONDITIONFair	-0.180788	0.195487	-0.925
REMODEL.age:CONDITIONGood	0.128282	0.039125	3.279
REMODEL.age:CONDITIONVery Good	1.319853	0.090222	14.629
ROOMS:AYB.age	0.005704	0.003354	1.701
ROOMS:ACY	0.937902	0.217902	4.304
BATHRM:CONDITIONExcellent	28.054394	1.329934	21.095
BATHRM:CONDITIONFair	2.305573	2.083741	1.106
BATHRM:CONDITIONGood	3.230037	0.491535	6.571
BATHRM:CONDITIONVery Good	18.855757	0.762262	24.737

Table #2 Credible Intervals

	2.5%	25%	50%	75%	97.5%
beta[1]	3.353	4.002	4.338	4.662	5.312
beta[2]	-0.434	0.177	0.510	0.830	1.417
beta[3]	0.984	1.349	1.535	1.718	2.101
beta[4]	-4.128	-3.545	-3.250	-2.964.	-2.398
beta[5]	7.605	7.974	8.172	8.357	8.729
beta[6]	0.496	0.532	0.548	0.566	0.600
beta[7]	-0.513	-0.482	-0.466	-0.451.	-0.422
beta[8]	-0.360	-0.322	-0.302	-0.282.	-0.243
beta[9}	-4.659	-4.170	-3.907	-3.659.	-3.209
beta[10]	-11.922	-9.942	-8.811	-7.765.	-5.605
beta[11]	-139.175	-49.370	-1.946	44.886	138.089
beta[12]	-138.019	-44.834	2.001	49.420	139.225
beta[13]	-0.029	-0.025	-0.023	-0.021.	-0.017
beta[14]	1.157	1.415	1.560	1.699	1.965
beta[15]	2.369	2.536	2.620	2.709	2.875

Table #3 Non-Convergence Diagnostic Chains

********** The Geweke diagnostic: **********

Z-scores:

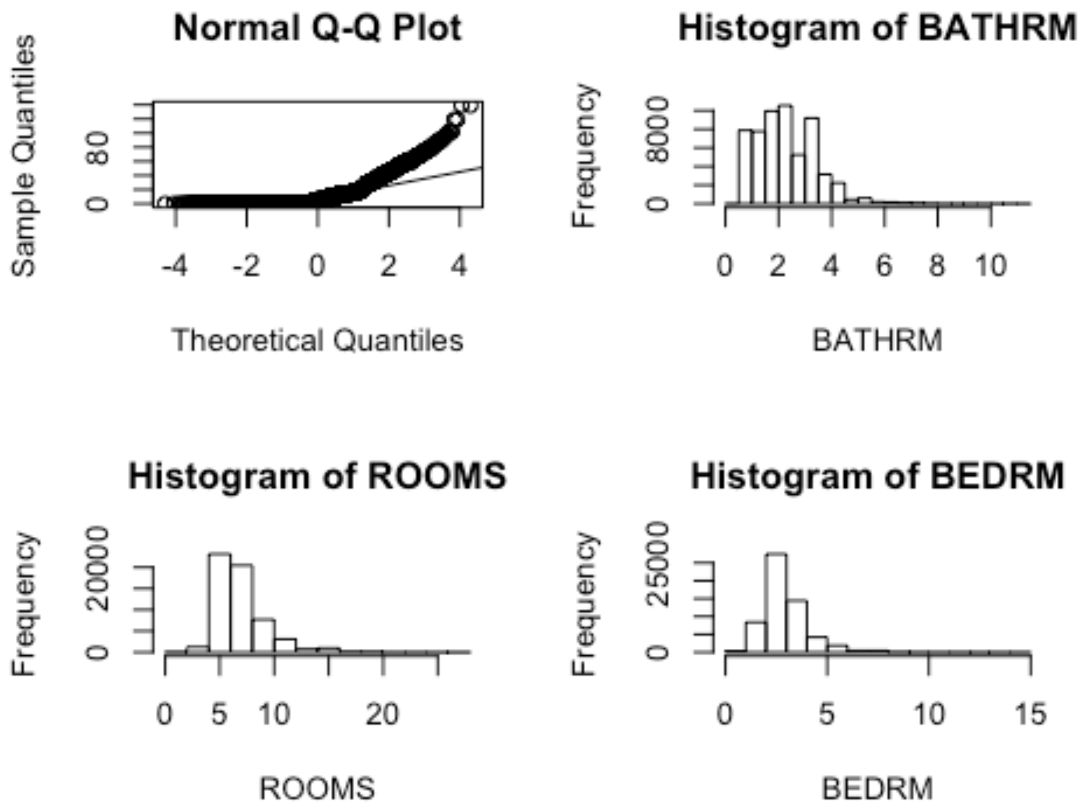
	chain1	chain 2	chain 3
beta[1]	-0.74844968	0.986269325	-1.35778951
beta[10]	0.64477438	0.742205620	-0.77222168
beta[11]	0.41618881	0.001370598	1.73010918
beta[12]	1.44043834	-1.495000447	-0.61371560
beta[13]	1.12663675	1.198251083	0.93890455
beta[14]	-0.19158933	-0.746746969	1.69220578
beta[2]	-1.89788451	0.351634285	0.35099685
beta[3]	0.27672861	0.455600910	-0.01112909
beta[4]	0.09220130	0.693058458	0.05611348
beta[5]	0.64501005	-0.582036524	1.21117122
beta[6]	-1.11022511	1.037259645	0.15686668
beta[7]	-0.08547827	1.709692319	1.97838990
beta[8]	0.96768930	-0.803961658	0.71389916
beta[9]	-0.41641887	-0.001344809	-1.73019474
Window From Start	0.10000000	0.890400000	0.75645000
Window From Stop	0.50000000	0.087150000	0.20965000

pD = 22.0 and DIC = 286221.4

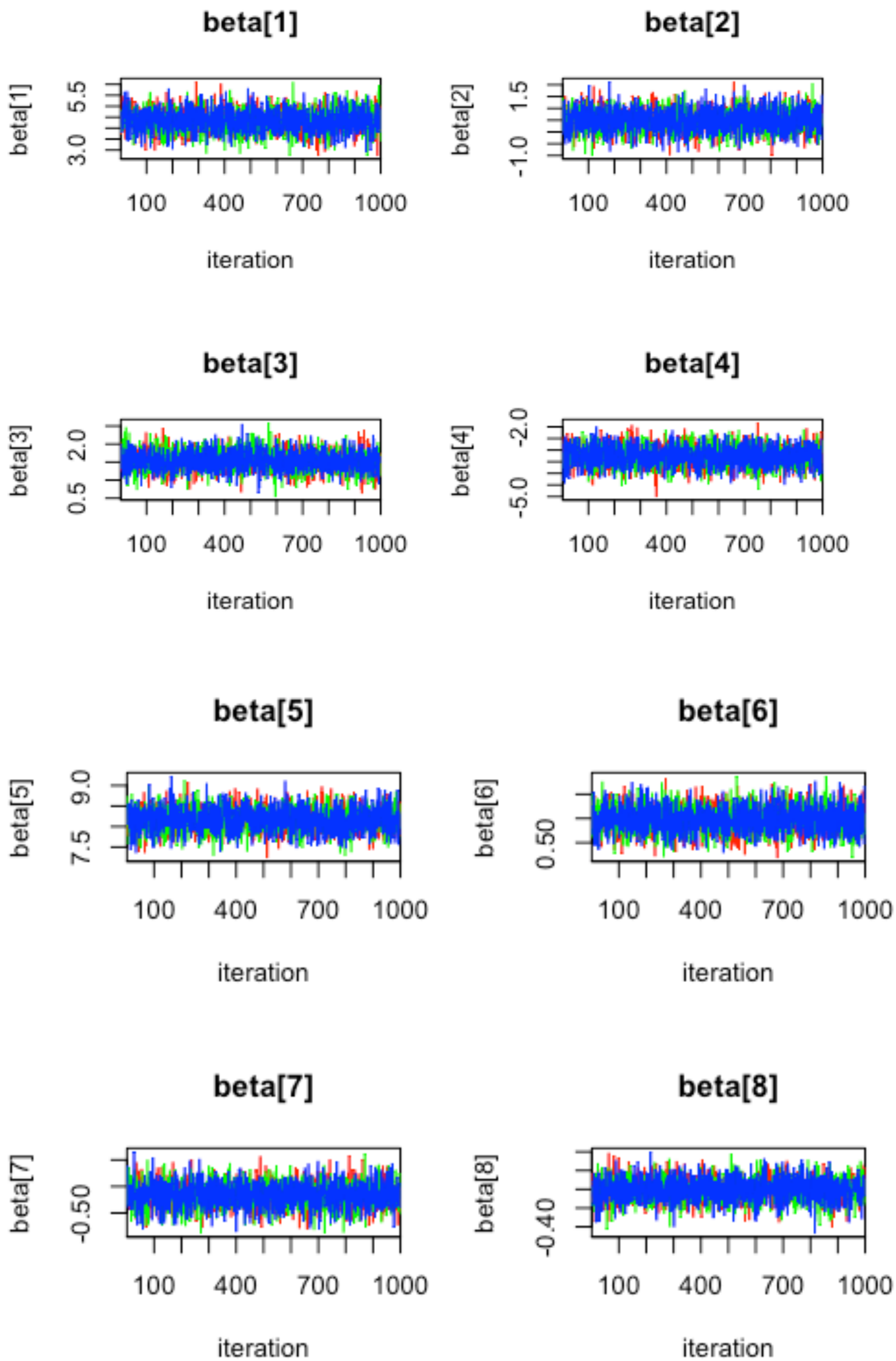
Table #4 Other Statistical Models

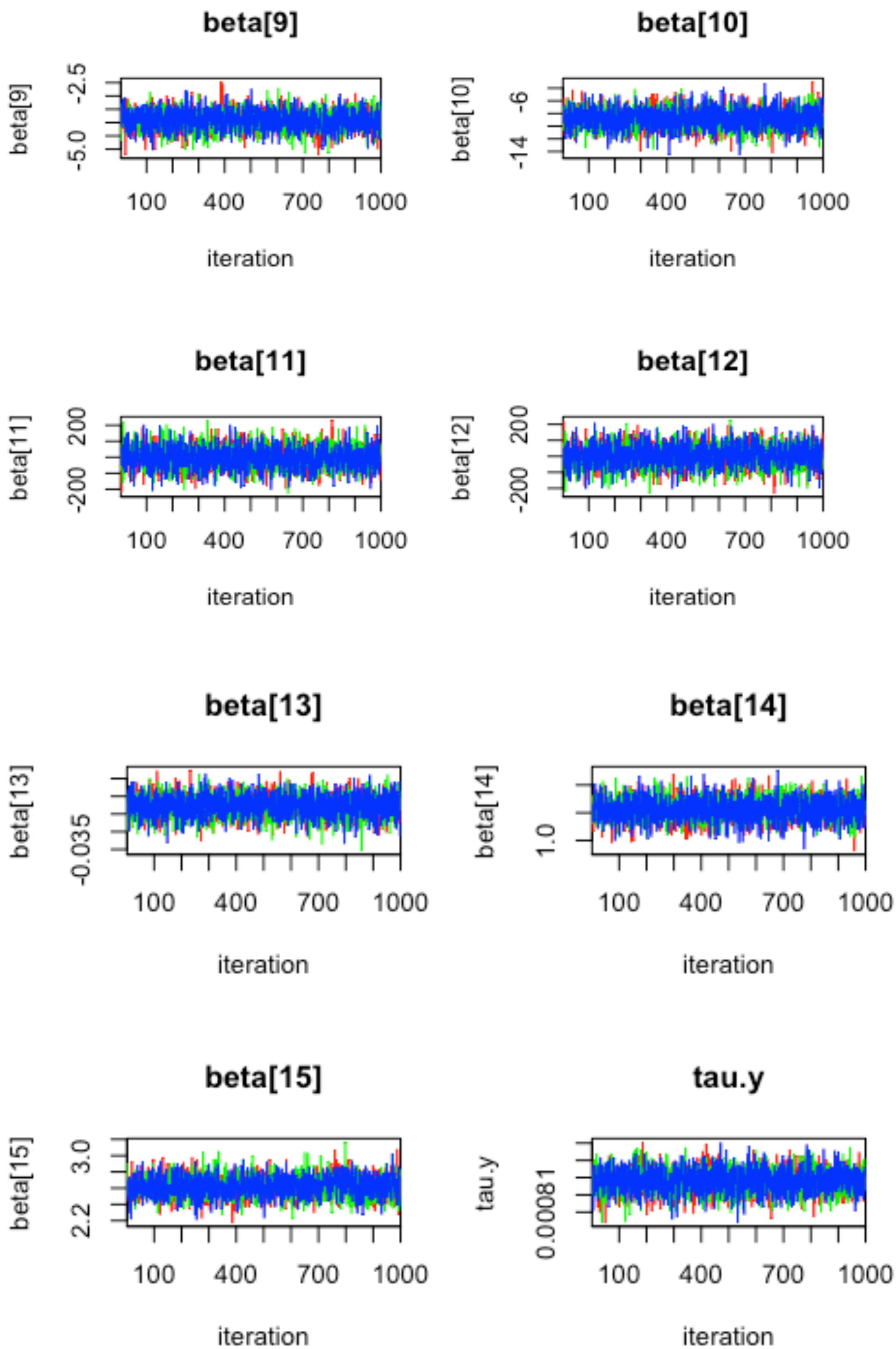
Logistic Regression	*Log(π/ 1-π)* = - 3.158 - 0.000004374*Price + 0.007901*√Price - 0.2907*(AC=Yes) + 0.1821*Rooms + 2.221*(Rooms^0.2) - 0.04816*√BEDRM + 0.1069*(CNDTN=Excellent) - 1.196*(CNDTN=Fair) + 0.2849*(CNDTN=Good) – 1.373*(CNDTN=Poor) + 0.6739(CNDTN=Very Good) – 0.4306*(Ward=2) – 0.2858*(Ward=3) – 0.0515*(Ward=4) + 0.2901(Ward=5) + 0.000001085PRICE*(AC=Yes) + 0.00000002.698*(PRICE*ROOMS) + 0.0000003.897(Price*Ward=2) + 0.0000005486*(Price*Ward=3) + 0.000001052*(Price*Ward=4) – 0.0000003.313*(Price*Ward=5)
Simple Linear Model	*Price_10K* = -7863 + 9.988*Bathroom + 0.7407*Rooms + 2.633*Bedrooms - 1.172*Stories - 1.003*Unqualified + 2.877*Grade=Average + 23.85*Grade=Excellent + 155.2*Grade=Exceptional + 15.33*Grade=Fair + 0.8134*Grade=Good + 46.50*Grade=Superior + 8.616*Grade=Very Good - 3.858*Kitchens + 9.305*Fireplaces + 8.452*Ward 2 + 12.01*Ward 3 - 4.366*Ward 4 - 5.517*Ward 5 + 3.526*Ward 6 -6.267*Ward 7 – 18.40*Ward 8 - 43.39*Latitude – 124*Latitude + 0.2879*AYB.age - 0.2937*EYB.age – 0.1789*Remodel.age + 41.17*Condition=Excellent + 6.386*Condition=Fair + 8.411*Condition=Good + 2.722*Condition=Very Good

Graph #1 Some Data Diagnostics



Graph #2 Trace plots





beta[9]
beta[9]
-2.5
-5.0
100
400
700
1000
iteration
beta[10]
beta[10]
-6
-14
100
400
700
1000
iteration
beta[11]
beta[11]
200
-200
100
400
700
1000
iteration
beta[12]
beta[12]
200
-200
100
400
700
1000
iteration
beta[13]
beta[13]
-0.035
100
400
700
1000
iteration
beta[14]
beta[14]
1.0
100
400
700
1000
iteration
beta[15]
beta[15]
3.0
2.2
100
400
700
1000
iteration
tau.y
tau.y
0.00081
100
400
700
1000
iteration

Analysis Code Appendix

Packages

```
library(tidyverse)
lapply(c("rjags","arm","coda","superdiag","R2WinBUGS","R2jags","lme4"), libra
ry,
       character.only=TRUE)
```

Data

```
DC_Properties <- read.csv("~/Documents/STAT 627 Statistical Machine Learning/
Stat 627 Project/Data/DC_Properties.csv")
DC_Properties <- data.frame(DC_Properties) # list to data.frame fix

## Final Cleaned Dataset

DC_Properties[is.na(DC_Properties)] <- 0 # setting all NA to 0
# makes data easier to clean and work with

select <- dplyr::select

DC_Properties_tidy <- DC_Properties %>%
  select(PRICE, BATHRM, HF_BATHRM, HEAT, AC, ROOMS, BEDRM, AYB, YR_RMDL, EYB,
STORIES, QUALIFIED, GRADE, CNDTN, KITCHENS, FIREPLACES, WARD, QUADRANT, LATIT
UDE, LONGITUDE) %>%
  mutate(PRICE = as.numeric(PRICE),
         BATHRM = as.numeric(BATHRM),
         HF_BATHRM = as.numeric(HF_BATHRM),
         ROOMS = as.numeric(ROOMS),
         BEDRM = as.numeric(BEDRM),
         AYB = as.numeric(AYB),
         EYB = as.numeric(EYB),
         STORIES = as.numeric(STORIES),
         KITCHENS = as.numeric(KITCHENS),
         FIREPLACES = as.numeric(FIREPLACES),
         LATITUDE = as.numeric(LATITUDE),
         LONGITUDE = as.numeric(LONGITUDE),
         HEAT = as.character(HEAT), # made character columns here to filter a
nd clean data better
         AC = as.character(AC),
         QUALIFIED = as.character(QUALIFIED),
         GRADE = as.character(GRADE),
         CNDTN = as.character(CNDTN)) %>%
  filter(CNDTN != "",
         CNDTN != "Default",
         CNDTN != "Poor",
         GRADE != " No Data",
         GRADE != "",
         HEAT != "No Data", # lose ~50,000 observations of the data by here
         PRICE > 10000 & PRICE < 10000000,
```

```
         FIREPLACES < 10,
         KITCHENS <= 10,
         ROOMS <= 40,
         BEDRM <= 20,
         STORIES <= 10,
         LATITUDE != 0,
         LONGITUDE != 0) %>% # Up to here, I lose roughly 100K observations
  mutate(AYB.age = AYB, # making new quantitative
         AYB.age = 2019 - AYB.age,
         AYB.age = ifelse(AYB == 2019, 0, AYB.age),
         EYB.age = as.numeric(EYB),
         EYB.age = 2019 - EYB.age,
         EYB.age = ifelse(EYB == 2019, 0, EYB.age),
         REMODEL.age = as.character(YR_RMDL),
         REMODEL.age = ifelse(REMODEL.age == "0", 0, REMODEL.age),
         REMODEL.age = ifelse(REMODEL.age == "20", 0, REMODEL.age),
         REMODEL.age = as.numeric(REMODEL.age),
         REMODEL.age = 2019 - REMODEL.age,
         REMODEL.age = ifelse(REMODEL.age == 2019, 0, REMODEL.age),
         GRADE = ifelse(GRADE == "Exceptional-A", "Exceptional", GRADE), # fi
xing Grade
         GRADE = ifelse(GRADE == "Exceptional-B", "Exceptional", GRADE),
         GRADE = ifelse(GRADE == "Exceptional-C", "Exceptional", GRADE),
         GRADE = ifelse(GRADE == "Exceptional-D", "Exceptional", GRADE),
         QUALIFIED_2 = QUALIFIED, # making new Qualified Variable
         QUALIFIED_2 = ifelse(QUALIFIED == "Q", 2, 1),
         QUALIFIED_2 = as.factor(QUALIFIED_2),
         AC = ifelse(AC == "0", "N", AC),
         GRADE = as.factor(GRADE), # fixed certain variables back to factors
         HEAT = as.factor(HEAT),
         AC = as.factor(AC),
         WARD = as.factor(WARD),
         CONDITION = as.factor(CNDTN),
         QUALIFIED = as.factor(QUALIFIED),
         AYB.age = as.numeric(AYB.age),
         EYB.age = as.numeric(EYB.age),
         REMODEL.age = as.numeric(REMODEL.age))%>%
  select(-CNDTN, -AYB, -YR_RMDL, -EYB) # select again

DC_Final <- na.omit(DC_Properties_tidy)

DC_Final <- DC_Final %>%
  filter(AYB.age < 2000) %>%
  mutate(PRICE_10K = PRICE/10000,
         BATHRM = BATHRM + HF_BATHRM*.5,
         BATHRM = as.numeric(BATHRM),
         GRADE2 = as.integer(GRADE)) %>%
  select(PRICE, PRICE_10K, BATHRM, ROOMS, BEDRM, STORIES, KITCHENS, FIREPLACE
S, LATITUDE,
         LONGITUDE, AYB.age, EYB.age, REMODEL.age, HEAT, AC, QUALIFIED, QUALI
```

```
FIED_2, GRADE,
         WARD, QUADRANT, CONDITION, GRADE2)

set.seed(10000000)

DC_Final <- sample_n(DC_Final, 57610, replace = TRUE) # want the datasets to
be even
# I lost one observation by doing this
dim(DC_Final)
summary(DC_Final)

## random case number

n <- length(DC_Final$PRICE)
Z <- sample(n,n/2)

bayes.train <- DC_Final[Z,] %>%
  as.list()
bayes.test <- DC_Final[-Z,] %>%
  as.list()
```

Final Model with a training set

```
## Model with Grade Grouping - best grouping I could find

final.bayes.model  <- function()  { # model name
    for (i in 1:N.dim) { # 1st for loop
      mu[i] <- alpha[GRADE[i]] + beta[1]*BATHRM[i] + beta[2]*ROOMS[i] + beta[
3]*BEDRM[i] +
        beta[4]*KITCHENS[i] + beta[5]*FIREPLACES[i] + beta[6]*AYB.age[i] + be
ta[7]*EYB.age[i] +
        beta[8]*REMODEL.age[i] + beta[9]*CONDITION[i] + beta[10]*AC[i] +
        beta[11]*REMODEL.age[i]*CONDITION[i] + beta[12]*REMODEL.age[i]*CONDIT
ION[i] +
        beta[13]*ROOMS[i]*AYB.age[i] + beta[14]*ROOMS[i]*AC[i] + beta[15]*CON
DITION[i]*BATHRM[i]
      PRICE_10K[i] ~ dnorm(mu[i],tau.y) # response variable distribution
      e.y[i] <- PRICE_10K[i] - mu[i]
    }
    beta[1]   ~ dnorm(0,0.0001); # beta distribution
    beta[2]   ~ dnorm(0,0.0001);
    beta[3]   ~ dnorm(0,0.0001);
    beta[4]   ~ dnorm(0,0.0001);
    beta[5]   ~ dnorm(0,0.0001);
    beta[6]   ~ dnorm(0,0.0001);
    beta[7]   ~ dnorm(0,0.0001);
    beta[8]   ~ dnorm(0,0.0001);
    beta[9]   ~ dnorm(0,0.0001);
    beta[10]   ~ dnorm(0,0.0001);
    beta[11]   ~ dnorm(0,0.0001);
```

```
    beta[12]   ~ dnorm(0,0.0001);
    beta[13]   ~ dnorm(0,0.0001);
    beta[14]   ~ dnorm(0,0.0001);
    beta[15]   ~ dnorm(0,0.0001);
    tau.y   ~ dgamma(1,0.1); # tau distributions

    s.y <- sd(e.y[])

    for (j in 1:N.GRADE) { # 2nd for loop
      alpha[j] ~ dnorm(0,tau.alpha) # grouping variable for newpid distributi
on
    }
    tau.alpha ~ dgamma(1,0.1);
}

train_jags <- bayes.train
train_jags["N.dim"] <- length(bayes.train[[1]])
train_jags["N.GRADE"] <- length(unique(bayes.train$GRADE2))

# SETUP INITIAL VALUES AND PARAMETER NAMES
grade.inits <- function(){
  list("tau.y" = 1, "tau.alpha" = 1, "beta" = rep(1,15))
}
grade.params <- c("beta", "tau.y", "tau.alpha", "s.y")

# RUN THE SAMPLER AND COLLECT CODA SAMPLES
final.model <- jags(data=train_jags, inits = grade.inits, grade.params, n.ite
r=10000, model=final.bayes.model, DIC = TRUE)
print(final.model) # DIC = 284692.4

asap.out.final <- update(final.model, n.iter=15000)
print(asap.out.final)
asap.mcmc.final <- as.mcmc(asap.out.final)
superdiag(as.mcmc.list(asap.mcmc.final), burnin=0)
```

Diagnostics

```
# Basic Model Diagnostics
traceplot(final.model, mfrow = c(1,1), varname = c("beta", "tau.y", "tau.alph
a", "s.y"), ask = FALSE)

# Data Diagnostics
attach(DC_Final)
par(mfrow=c(2,2))
qqnorm(BATHRM)
qqline(BATHRM)
qqnorm(ROOMS)
qqline(ROOMS)
qqnorm(BEDRM)
qqline(BEDRM)
qqnorm(STORIES)
```

```
qqline(STORIES)
qqnorm(KITCHENS)
qqline(KITCHENS)
qqnorm(FIREPLACES)
qqline(FIREPLACES)
qqnorm(AYB.age)
qqline(AYB.age)
qqnorm(EYB.age)
qqline(EYB.age)
qqnorm(REMODEL.age)
qqline(REMODEL.age)
hist(BATHRM)
hist(ROOMS)
hist(BEDRM)
hist(STORIES)
hist(KITCHENS)
hist(FIREPLACES)
hist(AYB.age)
hist(EYB.age)
hist(REMODEL.age)
par(mfrow=c(1,1))
detach(DC_Final)
```