**DOCUMENTATION FOR PROPERTY PARCEL DATA FROM THE TAX ASSESSOR’S DATABASE**

**Overview**

This document describes the structure and organization of the City of Boston Assessing Department’s centralized database for parcel-specific data for all uniquely identifiable parcels in the city (n = 165,661) for the year 2015. The data is released by the Assessing Department annually as part of the City of Boston’s open data initiative. Boston’s Assessing Department is responsible for determining accurate values for all properties in the city. To this end the Department maintains parcel ownership and value information to ensure fair assessment of both taxable and non-taxable property in Boston. This dataset can be used to analyze valuations for all classifications and types of property.

This dataset (*Tax Assesor 2015 - Data.csv)* is comprised of 54 City-generated variables related to parcel ownership, composition, and valuation information. In addition, this dataset contains 6 additional variables relating to each parcels location identification information including latitude and longitudinal coordinates, and 2010 Census block, group, and tract identifiers. An additional set of variables have been generated through analysis of the original dataset that provide other property identification information relating to both tax values and parcel-level composition characteristics.

*Table of Contents*

1. Summary of Assessing Department Data 3

1.1. Notes About this Amended Dataset 3

1.2. Assessing Department Variables 3

2. Geolocation and 2010 Census Data 8

2.1. Summary of Geolocation and 2010 Census Data 8

2.2. Description of Geolocation and 2010 Census Variables 8

3. Appendix 9

3.1. Appendix A: R-Syntax Code for Aggregating Parcels Files to the Parcel\_Num Level 9

3.2. Appendix B: R-Syntax Code for Adding Geographic Data from Parcels Files 10

3.3. Appendix C: R-Syntax Code for Adding Geographic Data from Geocoding 12

1. Summary of Assessing Department Data

The City of Boston’s Assessing Department is responsible for determining accurate values for all properties in the city. To this end the Department maintains parcel ownership and value information to ensure fair assessment of both taxable and non-taxable property in Boston. Assessing records are compiled and reviewed annually to reflect changes to parcels as a result of new construction, remodeling, and changes in ownership. The data contained within describes the parcel-specific address, ownership, type, structure, class, and valuation data. Upon annual review and re-assessment, parcel-specific data is updated and changes in calculated values are adjusted to reflect the most up to date status for each parcel. Property taxes (as indicated within this dataset as “GROSS\_TAX”) are also adjusted annually to reflect the annual taxation rates for residential and commercial properties.

Tax rates are determined by taking the portion of property tax levy for properties deemed “residential” and dividing that by the total assessed value of all residential properties in the City. Commercial tax rates are calculated in the same manner.[[1]](#footnote-1)

* 1. Notes About this Amended Dataset

This dataset is a modified version of the original Tax Assessor’s Data Set, and incorporates several additional variables and measures that were generated from manipulations to the original data. These new variables and measures provide additional insight into the current physical makeup of properties in Boston.

This dataset was also used to conduct a measure of the historical life of structures that are residential in type, and contain certain characteristics that indicate the historical evolution of a structure's life as well as analysis of structure types' actual locations within the city as they relate to Census demographics

* 1. Assessing Department Variables
* *PID* is the 10 digit parcel identification number, unique to each parcel. The first two digits indicate the Ward, digits 3 thru 7 are the parcel, and digits 8 thru 10 are the sub-parcel.
* *CM\_ID* is the 10 digit parcel number of the main condo building parcel. All condo units in each building are related to this number.
* *ST\_NUM* is the street number of the parcel.
* *ST\_NAME* is the street name of the parcel.
* *ST\_NAME\_SUF* is the suffix of the street name. This variable contains two-character short-forms of each type of suffix (St, Av, BL, PL, etc…)
* *UNIT\_NUM* is the specific unit number within a multi-unit building.
* *ZIPCODE* is the zip code of the parcel.
* *PTYPE* is the Property Type of the parcel. Property Types are classified according to State Class Codes, which are three digit codes. PTYPES between 001 and 299 are Residential properties. PTYPES between 300 and 399 are Commercial properties. PTYPES between 400 and 465 are Industrial properties. PTYPES greater than 900 are Exempt properties, meaning they are fully (or partially) exempt from property taxes.
* *LU* is the Land Use type for the parcel. R1 is Residential 1-Family, R2 is Residential 2-Family, R3 is Residential 3-Family, R4 is Residential 4 or more family, RL is Residential Land, A is Residential with 7 or more units, RC is Mixed Use (residential and commercial), CM is Condominium Main (the physical structure that houses all related condominium units), CD is a Residential Condo Unit, CP is Condominium Parking, CC is a Commercial Condominium, AH is Agricultural and Horticultural, C is Commercial, CL is Commercial Land, I is Industrial, E is Tax-Exempt, and EA is Tax-Exempt 121A properties (properties that are subject to MGLCH 121A, Section 10)
* *OWN\_OCC* is a one character code that indicates if an owner receives a residential exemption for the property. A "Y" indicates that the owner claims to live within the property (a.k.a. the property is "owner-occupied") and a "N" indicates the opposite.
* *OWNER* is the primary owner of the parcel, as of the date of the prior calendar year.
* *MAIL\_ADDRESS* is the street mailing address of the owner, to which the property tax bill is mailed.
* *MAIL.CS* is the city and state to which the property tax bill is mailed.
* *MAIL\_ZIPCODE* is the zip code of the property where the tax bill is mailed.
* *AV\_LAND* is the assessed value of the land.
* *AV\_BLDG* is the total assessed value for the building on the parcel.
* *AV\_TOTAL* is the total assessed value for the parcel. It is a summation of the assessed values of the land and building.
* *GROSS\_TAX* is the amount billed to the owner as property excise tax. It is based on the total assessed value multiplied by the tax rate. Tax rates are adjusted each year for Residential and Commercial property types.
* *LAND\_SF* is the total size of the parcel in square feet. This is also known as the lot size.
* *YR\_BUILT* is the year in which the property was built.
* *YR\_REMOD* is the year in which the property was last remodeled.
* *GROSS\_AREA* is the gross floor area for commercial properties.
* *LIVING\_AREA* is the total living area for residential properties.
* *NUM\_FLOORS* is the number of levels in the structure that is located on the parcel.
* *STRUCTURE\_CLASS* is the structural classification of commercial buildings. Classes include: *A* for Struct. Steel, *B* for Reinforced Concrete, *C* for Brick & Concrete, *D* for Wood/Frame, *E* for Metal, and *R* for Residential.
* *R\_BLDG\_STYL* is the building style for residential properties. The styles are: BL for Bi-Level, BW for Bungalow, CL for Colonial, CN for Contemporary, CP for Cape, CV for Conventional, DK for Decker, DX for Duplex, L for Tri-Level, Oth for Other, RE for Row End, RM for Row Middle, RN for Ranch, RR for Raised Ranch, SL for Split Level, TF for Two-Family Stack, TD for Tudor, SD for Semi\_Detatched, and VT for Victorian.
* *R\_ROOF\_TYPE* is the roof type for residential structures. Types include: F for Flat, G for Gable, H for Hip, L for Gambrel, M for Mansard, O for Other, and S for Shed.
* *R\_EXT\_FIN* is the exterior finish type for residential properties. Types include: A for Asbestos, B for Brick/Stone, C for Cement Board, F from Frame/Clapboard, G for Glass, K for Concrete, M for Vinyl, O for Other, P for Asphalt, S for Stucco, U for Aluminum, V for Brick/Stone Veneer, and W for Wood Shake.
* *R\_TOTAL\_RMS* is the total number of rooms in a residential structure.
* *R\_BDRMS* is the total number of bedrooms in a residential structure.
* *R\_FULL\_BATH* is the total number of Full Bathrooms in a residential structure. A full bath is also known as a four-piece bath, which includes a shower, a tub, a sink, and a toilet.
* *R\_HALF\_BATH* is the number of Half Bathrooms in a residential structure. A half bath typically includes a sink and a toilet, also known as a powder-room.
* *R\_KITCH* is the total number of kitchens in a residential structure.
* *R\_HEAT\_TYP* is the type of heating in a residential structure. Types include: N for None, S for Space Heater, W for Hot Water, E for Electric, P for Heat Pump, F for Forced Air, and O for Other.
* *R\_AC* indicates if the residential structure has air conditioning. Types include: C for Central Air Conditioning, D for Ductless Air Conditioning, and N for None.
* *R\_FPLACE* is the total number of fireplaces in a residential structure.
* *S\_NUM\_BLDG* is the number of buildings in condo main.
* *S\_BLDG\_STYLE* is the building style of of a Condo Main. Styles include: DK for Decker, DX for Duplex, CV for Conventional, FS for Free-Standing, HR for High Rise, LR for Low Rise, MD for Mid Rise, RE for Row End, RM for Row Middle, OT for Other, SD for Semi-Dethatched, and TH for Townhouse.
* *S\_UNIT\_RES* is the number of residential units in a condo main.
* *S\_UNIT\_COM* is the number of commercial units in a condo main.
* *S\_UNIT\_RC* is the number of mixed-use units in a condo main.
* *S\_EXT\_FIN* is the exterior finish type for condominium buildings. Types include: A for Asbestos, B for Brick & Stone, C for Cement Board, F for Frame/Clapboard, G for Glass, K for Concrete, M for Vinyl, O for Other, P for Asphalt, S for Stucco, U for Aluminum, V for Brick/Stone veneer, and W for Wood Shake.
* *U\_BASE\_FLOOR* indicates the base floor number of a condo unit. This would indicate if a condo unit is greater than one floor.
* *U\_NUM\_PARK*
* *U\_CORNER* indicates if a condo unit is located in the corner of a building. Y indicates Yes and N indicates No.
* *U\_ORIENT* indicates the orientation of the condo unit within the building. Types include: A for Rear Above, B for Rear Below, C for Face Courtyard, E for End, F for Front/Side, M for Middle, T for Through
* *U\_TOT\_ROOMS* is the total number of rooms in a condominium unit.
* *U\_BDRMS* is the total number of bedrooms in a condominium unit.
* *U\_FULL\_BTH* is the total number of Full Bathrooms in a condominium unit. A full bath is also known as a four-piece bath, which includes a shower, a tub, a sink, and a toilet.
* *­U\_HALF\_BTH is* the number of Half Bathrooms in a condominium unit. A half bath typically includes a sink and a toilet, also known as a powder-room.
* *U\_KIT\_TYP* is the type of kitchen in a condominium unit. Types include: F for Full Eat In, N for None, O for One-Person, and P for Pull/Alcove
* *U\_HEAT\_TYP* is the type of heating in a condominium unit. Types include: N for None, S for Space Heater, W for Hot Water, E for Electric, P for Heat Pump, F for Forced Air, and O for Other.
* *U\_AC* indicates if a condominium unit has self-controlled air conditioning. Types include: C for Central Air Conditioning, D for Ductless Air Conditioning, and N for None.
* *U\_FPLACE* is the number of fireplaces in a condominium unit.

1. Geolocation and 2010 Census Data
   1. Summary of Geolocation and 2010 Census Data

Geolocation information for parcels is provided in this dataset as additional identifying data that can improve analysis of Boston’s properties. The latitude and longitude is provided for each unique parcel and allows the user to perform spatial analysis using software, such as Esri’s ArcGIS suite. The coordinates are also useful in mapping.

The 2010 Census Data information is provided for each unique parcel as a way of grouping parcels using a recognized set of attributes. The Census Block, Block Group, and Tract identifiers for each parcel are set by the government in order to better understand the demographic makeup of cities, towns, states, etc… This data is provided within this dataset for easy connection to published data generated during the 2010 Census in Boston. These additional population demographic datasets can be connected to this dataset for additional analysis and research. Combining parcel specific data with Census data allows the user to generate insight into property and person-specific demographics.

2.2. Description of Geolocation and 2010 Census Variables

* *X* is the geo-coded longitude of the parcel
* *Y* is the geo-coded latitude of the parcel
* *TLID* is the identifier for the segment of road containing the parcel
* *Blk\_ID* is the 2010 Census Block ID number
* *BDG\_ID\_10* is the 2010 Census Group ID number
* *CT\_ID\_10* is the 2010 Census Tract ID number
* *LocationID* is the unique identifier for the address of the parcel.
* *BRA\_PD* is the name of the Boston Redevelopment Authority Planning District in which the building is located.
* *NSA\_NAME* is the name of the Inspectional Service Department Neighborhood Statistical Area in which the building is located.

1. Appendix
   1. Appendix A: R-Syntax Code for Aggregating Parcels Files to the Parcel\_Num Level

parcels2015 <- read.csv("/Users/henrygomory/Documents/Research/BARI/General Geographic Files/Parcels2015\_final.csv",stringsAsFactors=FALSE)

parcels2014 <- read.csv("/Users/henrygomory/Documents/Research/BARI/General Geographic Files/Parcels Boston 2014 BARI.csv",stringsAsFactors=FALSE)

#check that Blk\_IDs weren't cut off

options(scipen=999)

parcels2015$Blk\_ID\_10[10]

parcels2014$Blk\_ID\_10[10]

# the following syntax is from from parcelsToAddresses.R

Mode <- function(x) {

ux <- unique(x)

ux[which.max(tabulate(match(x, ux)))]

}

# making the parcel geographic data be accessible at the parcel num level, so they can be merged onto the tax assessor data

by\_parcel\_2015 <- sqldf("select parcel\_num, max(Blk\_ID\_10) `Blk\_ID\_10`,max(BG\_ID\_10) `BG\_ID\_10`,

max(CT\_ID\_10) `CT\_ID\_10`, max(X) `X`,

max(Y) `Y`, max(LocationID) `LocationID`, max(TLID) `TLID`

from parcels2015 group by parcel\_num")

# When we aggregate parcels to addresses we take max for this type of aggregation (on LocationID, Blk\_ID, etc.)

# I took mode instead because I think it makes more sense

# I'm aggregating these differently from the others above because the SQL mode function wasn't working for strings

# I could convert to integers then back but this seems more straightforward

by\_parcel\_BRAPD <- aggregate(BRA\_PD ~ parcel\_num, data=parcels2015, FUN=Mode)

by\_parcel\_2015<-merge(by\_parcel\_2015,by\_parcel\_BRAPD,by="parcel\_num",all.x=TRUE)

by\_parcel\_NSA <- aggregate(NSA\_NAME ~ parcel\_num, data=parcels2015, FUN=Mode)

by\_parcel\_2015<-merge(by\_parcel\_2015,by\_parcel\_NSA,by="parcel\_num",all.x=TRUE)

rm(by\_parcel\_BRAPD,by\_parcel\_NSA)

# the same aggregation procedure is then repeated for the 2014 parcels

by\_parcel\_2014 <- sqldf("select parcel\_num, max(Blk\_ID\_10) `Blk\_ID\_10`,max(BG\_ID\_10) `BG\_ID\_10`,

max(CT\_ID\_10) `CT\_ID\_10`, max(X) `X`,

max(Y) `Y`, max(LocationID) `LocationID`, max(TLID) `TLID`

from parcels2014 group by parcel\_num")

by\_parcel\_BRAPD <- aggregate(BRA\_PD ~ parcel\_num, data=parcels2014, FUN=Mode)

by\_parcel\_2014<-merge(by\_parcel\_2014,by\_parcel\_BRAPD,by="parcel\_num",all.x=TRUE)

by\_parcel\_NSA <- aggregate(NSA\_NAME ~ parcel\_num, data=parcels2014, FUN=Mode)

by\_parcel\_2014<-merge(by\_parcel\_2014,by\_parcel\_NSA,by="parcel\_num",all.x=TRUE)

rm(by\_parcel\_BRAPD,by\_parcel\_NSA)

#now we append the two parcel files (aggregated to the parcel num level)

by\_parcel\_long = rbind(by\_parcel\_2015,by\_parcel\_2014)

#delete duplicates between the two files, taking the 2015 values because they are earlier in the data

by\_parcel\_geo <- by\_parcel\_long[!duplicated(by\_parcel\_long$parcel\_num),]

#checks how much geographic data we have in the parcels file

for (columnName in c("X","Y","LocationID","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which( is.na(by\_parcel\_geo[,columnName]) | by\_parcel\_geo[,columnName] == 0 ))) ) )

}

* 1. Appendix B: R-Syntax Code for Adding Geographic Data from Parcels Files

library(plyr)

check\_geo <- function(x) {

for (columnName in c("X","Y","LocationID","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which( is.na(x[,columnName]) | x[,columnName] == 0 ))) ) )

}

}

tax\_assessor = read.csv("/Users/henrygomory/Documents/Research/BARI/Course Data/Tax Assessor Cross/DATA FULL.txt",stringsAsFactors = FALSE)

library(sqldf)

parcels2015 <- read.csv("/Users/henrygomory/Documents/Research/BARI/General Geographic Files/Parcels2015\_final.csv",stringsAsFactors=FALSE)

parcels2014 <- read.csv("/Users/henrygomory/Documents/Research/BARI/General Geographic Files/Parcels Boston 2014 BARI.csv",stringsAsFactors=FALSE)

#check that Blk\_IDs weren't cut off

options(scipen=999)

parcels2015$Blk\_ID\_10[10]

parcels2014$Blk\_ID\_10[10]

# the following syntax is from from parcelsToAddresses.R

Mode <- function(x) {

ux <- unique(x)

ux[which.max(tabulate(match(x, ux)))]

}

# making the parcel geographic data be accessible at the parcel num level, so they can be merged onto the tax assessor data

by\_parcel\_2015 <- sqldf("select parcel\_num, max(Blk\_ID\_10) `Blk\_ID\_10`,max(BG\_ID\_10) `BG\_ID\_10`,

max(CT\_ID\_10) `CT\_ID\_10`, max(X) `X`,

max(Y) `Y`, max(LocationID) `LocationID`, max(TLID) `TLID`

from parcels2015 group by parcel\_num")

# When we aggregate parcels to addresses we take max for this type of aggregation (on LocationID, Blk\_ID, etc.)

# I took mode instead because I think it makes more sense

# I'm aggregating these differently from the others above because the SQL mode function wasn't working for strings

# I could convert to integers then back but this seems more straightforward

by\_parcel\_BRAPD <- aggregate(BRA\_PD ~ parcel\_num, data=parcels2015, FUN=Mode)

by\_parcel\_2015<-merge(by\_parcel\_2015,by\_parcel\_BRAPD,by="parcel\_num",all.x=TRUE)

by\_parcel\_NSA <- aggregate(NSA\_NAME ~ parcel\_num, data=parcels2015, FUN=Mode)

by\_parcel\_2015<-merge(by\_parcel\_2015,by\_parcel\_NSA,by="parcel\_num",all.x=TRUE)

rm(by\_parcel\_BRAPD,by\_parcel\_NSA)

# the same aggregation procedure is then repeated for the 2014 parcels

by\_parcel\_2014 <- sqldf("select parcel\_num, max(Blk\_ID\_10) `Blk\_ID\_10`,max(BG\_ID\_10) `BG\_ID\_10`,

max(CT\_ID\_10) `CT\_ID\_10`, max(X) `X`,

max(Y) `Y`, max(LocationID) `LocationID`, max(TLID) `TLID`

from parcels2014 group by parcel\_num")

by\_parcel\_BRAPD <- aggregate(BRA\_PD ~ parcel\_num, data=parcels2014, FUN=Mode)

by\_parcel\_2014<-merge(by\_parcel\_2014,by\_parcel\_BRAPD,by="parcel\_num",all.x=TRUE)

by\_parcel\_NSA <- aggregate(NSA\_NAME ~ parcel\_num, data=parcels2014, FUN=Mode)

by\_parcel\_2014<-merge(by\_parcel\_2014,by\_parcel\_NSA,by="parcel\_num",all.x=TRUE)

rm(by\_parcel\_BRAPD,by\_parcel\_NSA)

#now we append the two parcel files (aggregated to the parcel num level)

by\_parcel\_long = rbind(by\_parcel\_2015,by\_parcel\_2014)

#delete duplicates between the two files, taking the 2015 values because they are earlier in the data

by\_parcel\_geo <- by\_parcel\_long[!duplicated(by\_parcel\_long$parcel\_num),]

#checks how much geographic data we have in the parcels file

for (columnName in c("X","Y","LocationID","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which( is.na(by\_parcel\_geo[,columnName]) | by\_parcel\_geo[,columnName] == 0 ))) ) )

}

#merge the parcel geographic data onto the tax assessor data

tax\_assessor\_geo<-merge(tax\_assessor,by\_parcel\_geo,by.x="PID",by.y="parcel\_num",all.x=TRUE)

#checks how many observations in the tax assessor file were missing parcel nums

length(which(tax\_assessor\_geo$PID==0 | tax\_assessor\_geo$PID=="NULL" | is.na(tax\_assessor\_geo$PID)))

#shows all missing data

for (columnName in c("X","Y","LocationID","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which( is.na(tax\_assessor\_geo[,columnName]) | tax\_assessor\_geo[,columnName] == 0 ))) ) )

}

* 1. Appendix C: R-Syntax Code for Adding Geographic Data from Geocoding

extra\_geo = read.csv("/Users/henrygomory/Documents/Research/BARI/Course Data/missing addresses/ADDRESSES\_WITH\_GEO.csv",stringsAsFactors = FALSE)

#the geographic data file has data for multiple files, so we subset it to just licenses using type == L

extra\_geo = extra\_geo[extra\_geo$type=="TAC",c("ID","X","Y","TLID","GEOID10","BG\_ID\_10","CT\_ID\_10","NSA\_NAME","BRA\_PD")]

extra\_geo <- rename(extra\_geo, replace=c("GEOID10" = "Blk\_ID\_10"))

#check that the Blk ID wasn't truncated

head(extra\_geo[!is.na(extra\_geo$Blk\_ID\_10),"Blk\_ID\_10"])

#Use Blk ID and BG ID to fill in possibly missing CTs and BGs

sub\_1 <- function(x) {

return(substr(as.character(as.numeric(x)),1,12))

}

sub\_2 <- function(x) {

return(substr(as.character(as.numeric(x)),1,11))

}

Blk\_missing = extra\_geo[,"Blk\_ID\_10"]==0 | extra\_geo[,"Blk\_ID\_10"]=="NULL" | is.na(extra\_geo[,"Blk\_ID\_10"])

BG\_missing = extra\_geo[,"BG\_ID\_10"]==0 | extra\_geo[,"BG\_ID\_10"]=="NULL" | is.na(extra\_geo[,"BG\_ID\_10"])

CT\_missing = extra\_geo[,"CT\_ID\_10"]==0 | extra\_geo[,"CT\_ID\_10"]=="NULL" | is.na(extra\_geo[,"CT\_ID\_10"])

extra\_geo[ BG\_missing & !Blk\_missing , "BG\_ID\_10" ]= (as.character(lapply(extra\_geo$Blk\_ID\_10,sub\_1)))[BG\_missing & !Blk\_missing]

extra\_geo[ CT\_missing & !Blk\_missing , "CT\_ID\_10" ]=as.character(lapply(extra\_geo$Blk\_ID\_10,sub\_2))[CT\_missing & !Blk\_missing]

extra\_geo[ CT\_missing & !BG\_missing , "CT\_ID\_10" ]=as.character(lapply(extra\_geo$BG\_ID\_10,sub\_2))[CT\_missing & !BG\_missing]

#check how much geographic data we have

for (columnName in c("X","Y","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which(!(extra\_geo[,columnName]==0 | extra\_geo[,columnName]=="NULL" | is.na(extra\_geo[,columnName])))))))

}

#get rid of duplicates by License Number (every License Number has the same address attached so geographic data will be the same)

extra\_geo = extra\_geo[!duplicated(extra\_geo$ID),]

#merge in extra geographic info

tax\_assessor\_geo = merge(tax\_assessor\_geo,extra\_geo,by.x="PID",by.y="ID",all.x=TRUE)

tax\_assessor\_geo$X.x[!is.na(tax\_assessor\_geo$X.y)] = tax\_assessor\_geo$X.y[!is.na(tax\_assessor\_geo$X.y)]

tax\_assessor\_geo$Y.x[!is.na(tax\_assessor\_geo$Y.y)] = tax\_assessor\_geo$Y.y[!is.na(tax\_assessor\_geo$Y.y)]

tax\_assessor\_geo$TLID.x[!is.na(tax\_assessor\_geo$TLID.y)] = tax\_assessor\_geo$TLID.y[!is.na(tax\_assessor\_geo$TLID.y)]

tax\_assessor\_geo$Blk\_ID\_10.x[!is.na(tax\_assessor\_geo$Blk\_ID\_10.y)] = tax\_assessor\_geo$Blk\_ID\_10.y[!is.na(tax\_assessor\_geo$Blk\_ID\_10.y)]

tax\_assessor\_geo$BG\_ID\_10.x[!is.na(tax\_assessor\_geo$BG\_ID\_10.y)] = tax\_assessor\_geo$BG\_ID\_10.y[!is.na(tax\_assessor\_geo$BG\_ID\_10.y)]

tax\_assessor\_geo$CT\_ID\_10.x[!is.na(tax\_assessor\_geo$CT\_ID\_10.y)] = tax\_assessor\_geo$CT\_ID\_10.y[!is.na(tax\_assessor\_geo$CT\_ID\_10.y)]

tax\_assessor\_geo$NSA\_NAME.x[!is.na(tax\_assessor\_geo$NSA\_NAME.y)] = tax\_assessor\_geo$NSA\_NAME.y[!is.na(tax\_assessor\_geo$NSA\_NAME.y)]

tax\_assessor\_geo$BRA\_PD.x[!is.na(tax\_assessor\_geo$BRA\_PD.y)] = tax\_assessor\_geo$BRA\_PD.y[!is.na(tax\_assessor\_geo$BRA\_PD.y)]

tax\_assessor\_geo = rename(tax\_assessor\_geo, c("X.x"="X", "Y.x"="Y","TLID.x"="TLID","Blk\_ID\_10.x"="Blk\_ID\_10", "BG\_ID\_10.x"="BG\_ID\_10","CT\_ID\_10.x"="CT\_ID\_10","BRA\_PD.x"="BRA\_PD","NSA\_NAME.x"="NSA\_NAME"))

tax\_assessor\_geo = tax\_assessor\_geo[,setdiff(colnames(tax\_assessor\_geo),c("X.y","Y.y","TLID.y","Blk\_ID\_10.y","BG\_ID\_10.y","CT\_ID\_10.y","NSA\_NAME.y","BRA\_PD.y"))]

#check how much geographic data we have

for (columnName in c("X","Y","TLID","Blk\_ID\_10","BG\_ID\_10","CT\_ID\_10","BRA\_PD","NSA\_NAME")) {

print(paste((columnName),"-", (length(which(!(tax\_assessor\_geo[,columnName]==0 | tax\_assessor\_geo[,columnName]=="NULL" | is.na(tax\_assessor\_geo[,columnName]) ))))))

}

tax\_assessor\_geo = rename(tax\_assessor\_geo,c("Blk\_ID\_10"="Blk\_ID"))

write.csv(tax\_assessor\_geo,"/Users/henrygomory/Documents/Research/BARI/Course Data/Finished Files/Tax Assessor (2015 Clean).csv")

1. Tax rate calculation information published by the City of Boston Assessing Department through the department’s website: http://www.cityofboston.gov/assessing/taxrates.asp [↑](#footnote-ref-1)