NYC Real Estate Transactions: Code File

Quentin McTeer

5/02/2021

Contents

Preprocessing **1**

Create Analytic Dataset **2**

Exploratory Data Analysis **3**

Visualizations 3
Summary Statistics 5
Correlation Tables 5
Regression and Machine Learning Models 6

Preprocessing

```
library(tidyverse)
library(broom)
library(modelsummary)
library(kableExtra)
library(cowplot)
library(qwraps2)
library(gtsummary)
library(skimr)
library(readxl)
library(fixest)
library(tidymodels)
library(pROC)
library(corrplot)
library(Hmisc)
library(huxtable)
options(qwraps2_markup = "markdown")
options(scipen = 999) # load packages and set options

nyc <- read_csv("nyc-rolling-sales.csv")
inc <- read_xlsx("inc.xlsx")  # read in data
```

```
 # skim(nyc) # evaluate data structure and missing values
```

```
nyc <- nyc %>%
  rename("id" = X1,
         "borough" = BOROUGH,
         "neighborhood" = NEIGHBORHOOD,
         "bld_clss_cat" = `BUILDING CLASS CATEGORY`,
```

```
         "tax_clss_pres" = `TAX CLASS AT PRESENT`,
         "block" = BLOCK,
         "lot" = LOT,
         "easement" = `EASE-MENT`,
         "bld_clss_pres" = `BUILDING CLASS AT PRESENT`,
         "address" = ADDRESS,
         "apt_no" = `APARTMENT NUMBER`,
         "zip" = `ZIP CODE`,
         "resid_units" = `RESIDENTIAL UNITS`,
         "commer_units" = `COMMERCIAL UNITS`,
         "ttl_units" = `TOTAL UNITS`,
         "lnd_sqft" = `LAND SQUARE FEET`,
         "gross_sqft" = `GROSS SQUARE FEET`,
         "year_built" = `YEAR BUILT`,
         "tax_clss_sale" = `TAX CLASS AT TIME OF SALE`,
         "bld_clss_sale" = `BUILDING CLASS AT TIME OF SALE`,
         "price" = `SALE PRICE`,
         "date" = `SALE DATE`) # rename columns
```

Create Analytic Dataset

```
nyc <- left_join(x = nyc, y = inc, by = "borough") # merge in median income, crime rates, and percent w
```

```
nyc <- nyc %>%
  mutate(lnd_sqft = ifelse(lnd_sqft == "-", NA, lnd_sqft),
         gross_sqft = ifelse(gross_sqft == "-", NA, gross_sqft),
         price = ifelse(price == "-", NA, price),
         year_built = ifelse(year_built == 0, NA, year_built)) # create NA values for missing data

nyc <- nyc %>%
  select(-easement, -apt_no, -lot, -bld_clss_pres, -tax_clss_pres, -block, -zip) # eliminate easement a

nyc <- na.omit(nyc) # omit missing values, imputation won't make sense for many missing values in the d
```

```
nyc <- nyc %>%
  mutate(bld_clss_cat = as.factor(bld_clss_cat),
         lnd_sqft = as.numeric(lnd_sqft),
         gross_sqft = as.numeric(gross_sqft),
         bld_clss_sale = as.factor(bld_clss_sale),
         price = as.numeric(price),
         borough = as.factor(borough),
         tax_clss_sale = as.factor(tax_clss_sale),
         white = as.numeric(white)) # coerce columns to fit more accurate data structure
```

```
nyc %>%
  filter(price < 1500) %>%
  count() # check number of deed transfers

nyc <- nyc %>%
  mutate(deed_trans = ifelse(price < 1500, "TRUE", "FALSE" )) # generate deed transfer indicator variab
```

```
nyc %>%
  group_by(bld_clss_cat) %>%
```

```
  count() %>%
  arrange(desc(n))

nyc <- nyc %>%
  mutate(bld_clss_cat = fct_lump(bld_clss_cat, prop = 0.045)) # shrink building class category to seven
```

```
nyc <- nyc %>%
  filter(bld_clss_cat != "Other") # include only residential property transactions
```

```
nyc %>%
  group_by(bld_clss_sale) %>%
  count() %>%
  arrange(desc(n)) # check number of categories and frequency of each category

nyc <- nyc %>%
  mutate(bld_clss_sale = fct_lump(bld_clss_sale, prop = 0.04))  # shrink building class at time of sale
```

```
gen <- nyc %>%
  group_by(borough) %>%
  summarise(mean_sqft = mean(gross_sqft)) # create mean square feet by borough variable

nyc <- left_join(x = nyc, y = gen, by = "borough") # merge the variable into the original dataset
```

```
nyc <- nyc %>%
  mutate(borough = ifelse(borough == 1, "Manhattan", borough),
         borough = ifelse(borough == 2, "Bronx", borough),
         borough = ifelse(borough == 3, "Brooklyn", borough),
         borough = ifelse(borough == 4, "Queens", borough),
         borough = ifelse(borough == 5, "Staten Island", borough)) # edit borough variable to display b
```

```
nyc <- separate(nyc, date, c("year", "month", "day"), sep = "-") # create separate year, month and day

nyc$date <- paste(nyc$year, nyc$month, nyc$day, sep="/") # Recreate date variable

nyc$date <- as.Date(nyc$date) # Read date variable as date
```

Exploratory Data Analysis

Visualizations

```
nyc %>%
  filter(price < 5000000 & deed_trans == FALSE) %>%
  ggplot(nyc, mapping = aes(x = price)) +
  geom_histogram(bins = 12, fill = "#90BDDD", color = "white") +
  ylab("Number of Transactions") +
  xlab("Sale Price (in 000's)") +
  theme_cowplot(12) # plot sale price distribution
```

```
nyc %>%
  group_by(borough) %>%
  summarise(n = n()) %>%
  ggplot() +
  geom_bar(aes(x = borough, y = n), stat = "identity", fill = "#90BDDD") +
  ylab("Number of Transactions") +
```

```
  xlab("Borough") +
  theme_cowplot(12) # plot real estate sales by borough
```

```
nyc %>%
  filter(deed_trans == FALSE) %>%
  group_by(bld_clss_cat) %>%
  summarise(avg.price = mean(price)) %>%
  ggplot() +
  geom_bar(aes(x = avg.price/1000, y = bld_clss_cat), stat = "identity", fill = "#90BDDD") +
  ylab("Building Class Category") +
  xlab("Price (000's of $)") +
  theme_cowplot(12) # plot average sale price price by building type
```

```
nyc %>%
  filter(deed_trans == FALSE) %>%
  group_by(bld_clss_sale) %>%
  summarise(avg.price = mean(price)/1000) %>%
  ggplot() +
  geom_bar(aes(x = avg.price, y = bld_clss_sale), stat = "identity", fill = "#90BDDD") +
  ylab("Building Class at Sale") +
  xlab("Price in 000's in $") +
  theme_cowplot(12) # plot average sale price by building class at sale
```

```
nyc %>%
  filter(deed_trans == FALSE) %>%
  group_by(month) %>%
  count() %>%
  ggplot() +
  geom_point(aes(x = month, y = n)) +
  geom_line(aes(x = month, y = n, group = 1)) +
  geom_vline(aes(xintercept = "09"), color = "#90BDDD" ) +
  ylab("Number of Transactions") +
  xlab("Month") +
  theme_cowplot(12) # plot monthly trends in NYC real estate transactions
```

```
nyc %>%
  filter(deed_trans == FALSE & price < 5000000) %>%
  group_by(year_built) %>%
  summarise(avg.price = mean(price)) %>%
  ggplot() +
  geom_point(aes(x = year_built, y = avg.price)) +
  ylab("Average Sale Price") +
  xlab("Year Built") +
  geom_smooth(aes(x = year_built, y = avg.price), color = "#90BDDD", method = "lm") +
  theme_cowplot(12) # plot year built against sale price
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
nyc %>%
  group_by(tax_clss_sale) %>%
  summarise(avg.price = mean(price)) %>%
  ggplot() +
  geom_bar(aes(x = avg.price/1000, y = tax_clss_sale), stat = "identity", fill = "#90BDDD") +
  xlab("Average Sale Price (000's of $'s) ") +
  ylab("Building & Building Units: Tax Class") +
```

```
  theme_cowplot(12) # plot bar chart of tax class on sale price
```

```
nyc %>%
  filter(deed_trans == FALSE) %>%
  filter(gross_sqft > 0 & gross_sqft < 10000 & price < 10000000) %>%
  group_by(gross_sqft) %>%
  summarise(avg.price = mean(price)) %>%
  ggplot() +
  geom_point(aes(x = gross_sqft, y = avg.price/1000)) +
  theme_cowplot(12) +
  ylab("Average Sale Price (000's of $)") +
  xlab("Gross Square Feet") +
  geom_smooth(aes(x = gross_sqft, y = avg.price/1000), color = "#90BDDD", method = "lm") # plot gross s
```

```
## `geom_smooth()` using formula 'y ~ x'
```

Summary Statistics

```
nyc %>%
  select(deed_trans, resid_units, commer_units, ttl_units, lnd_sqft, gross_sqft, year_built, price, bor
  tbl_summary(
    by = deed_trans,
    statistic = list(all_continuous() ~ "{mean} ({sd})"),
    digits = all_continuous() ~ 2,
  ) %>%
  as_hux_table() %>%
  theme_plain # create summary statistics table segmented by deed transfer
```

```
tb_hux <- nyc %>%
  select(income, crime, white, resid_units, commer_units, ttl_units, lnd_sqft, gross_sqft, year_built,
  tbl_summary(
    by = borough,
    statistic = list(all_continuous() ~ "{mean}"),
    digits = all_continuous() ~ 2,
    type = list(income ~ 'continuous',
                crime ~ 'continuous',
                white ~ 'continuous')) %>%
  as_hux_table # create summary statistics table

tb_hux %>%
  set_width(0.9) %>%
set_font_size(10) %>%
  theme_plain() %>%
set_position("center") # report summary statistics table
```

Correlation Tables

```
res2 <- cor((nyc[,c("income", "crime", "white", "resid_units", "commer_units", "ttl_units", "lnd_sqft",

res2 %>%
  kbl(digits = 3, booktabs = T) %>%
  kable_styling(full_width = FALSE, position = "center",
                latex_options = c("striped", "hold_position")) # report correlation matrix
```

Regression and Machine Learning Models

```
nyc_pred <- nyc %>%
  filter(deed_trans == FALSE) %>%
  select(-neighborhood, -id, -address, -day, -date, -mean_sqft, -year,
         -deed_trans, -white, -ttl_units, -crime) %>%
  mutate(month = as.factor(month),
         borough = as.factor(borough)) # create prediction data set from analytic data set for linear r

borough_model <- feols(price ~ borough + resid_units +  commer_units + lnd_sqft
                       data = nyc_pred) # build causal inference model

tidy(borough_model, se = "white") %>%
  kbl(digits = 3, booktabs = T) %>%
  kable_styling(full_width = FALSE, position = "center",
                latex_options = c("striped", "hold_position")) # create table

glance(borough_model)

RSS <- c(crossprod(borough_model$residuals))
MSE <- RSS / length(borough_model$residuals)
RMSE <- sqrt(MSE)
RMSE # pull out metrics
```

```
set.seed(555)

nyc_split <- initial_split(nyc_pred)

nyc_train <- training(nyc_split)
nyc_test <- testing(nyc_split) # create test and training dataset

nyc_wf <- workflow() %>%
  add_formula(price ~ .)

rf_spec <- rand_forest(
  mtry = 6,
  trees = 1000,
 min_n = 5) %>%
 set_mode("regression") %>%
  set_engine(engine = "ranger")

 # rf_final <- nyc_wf %>%
 # add_model(rf_spec) %>%
# last_fit(nyc_split) # train model and gather test results

# rf_final

# rf_final$.workflow[[1]]

 #collect_metrics(rf_final) # report metrics
```

```
set.seed(1234)
nyc_bin <- nyc %>%
 select(-neighborhood, -id, -address, -day, -date, -mean_sqft, -year, -crime, -price, -income, -white) %
```

```
 mutate(month = as.factor(month),
        deed_trans = as.factor(deed_trans),
        borough = as.factor(borough)) # create dataset classification models

nyc_split <- initial_split(nyc_bin)

nyc_train1 <- training(nyc_split)
nyc_test1 <- testing(nyc_split) # create training/test data

 nyc_wf1 <- workflow() %>%
 add_formula(deed_trans ~ .)

 log_spec <- logistic_reg() %>%
  set_engine(engine = "glm") # build model

 # log_final <- nyc_wf1 %>%
 # add_model(log_spec) %>%
 # last_fit(nyc_split) # run model on train/test data

 # log_final

 # log_final$.workflow[[1]]

#  collect_metrics(log_final) # report metrics
```

```
 set.seed(542)
rf_spec1 <- rand_forest(
 mtry = 8,
 trees = 1000,
 min_n = 8) %>%
 set_mode("classification") %>%
 set_engine(engine = "ranger") # create model

 # rf_final1 <- nyc_wf1 %>%
 # add_model(rf_spec1) %>%
 # last_fit(nyc_split) # run model on training and test data

 # rf_final1

 # rf_final1$.workflow[[1]]

 # collect_metrics(rf_final1) # report metrics
```