Part 2

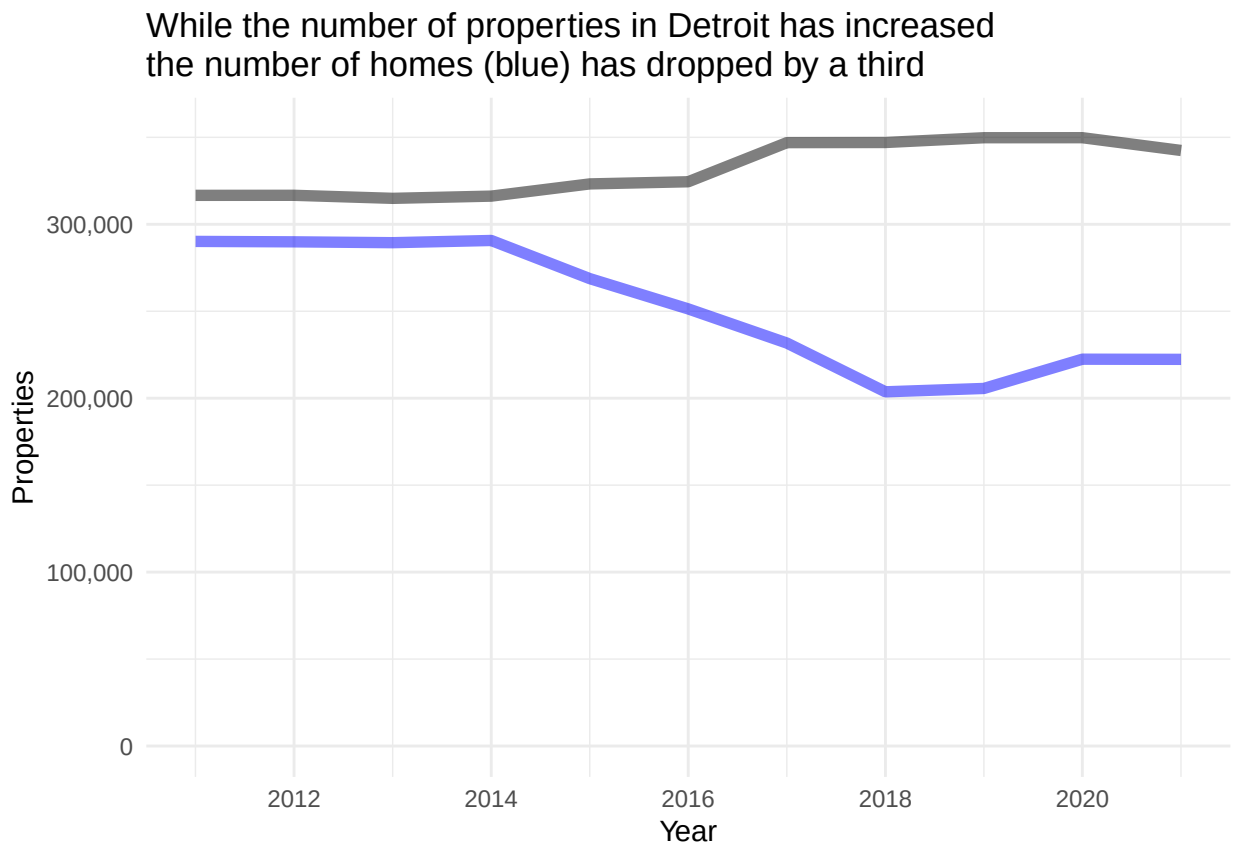
Ethan Jantz

2/27/2022

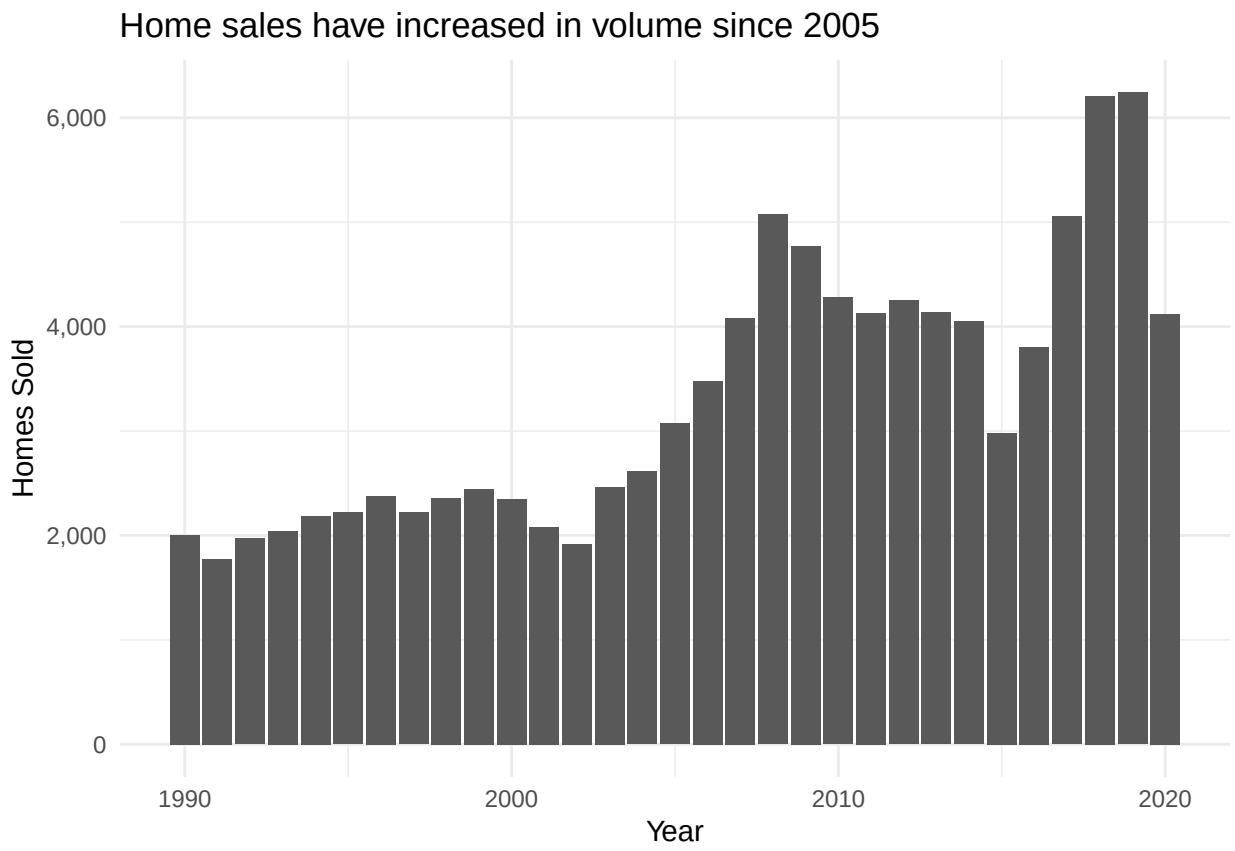
Introduction

Detroit's residential property stock has been in decline since 2014, which is around the time that home sales fell to their lowest point in the 2010's. This coincides with the highest rate of foreclosures in the city in the decade. Detroit aims to assess homes at 50% of their value, and while there has been progress toward this metric assessments continue to

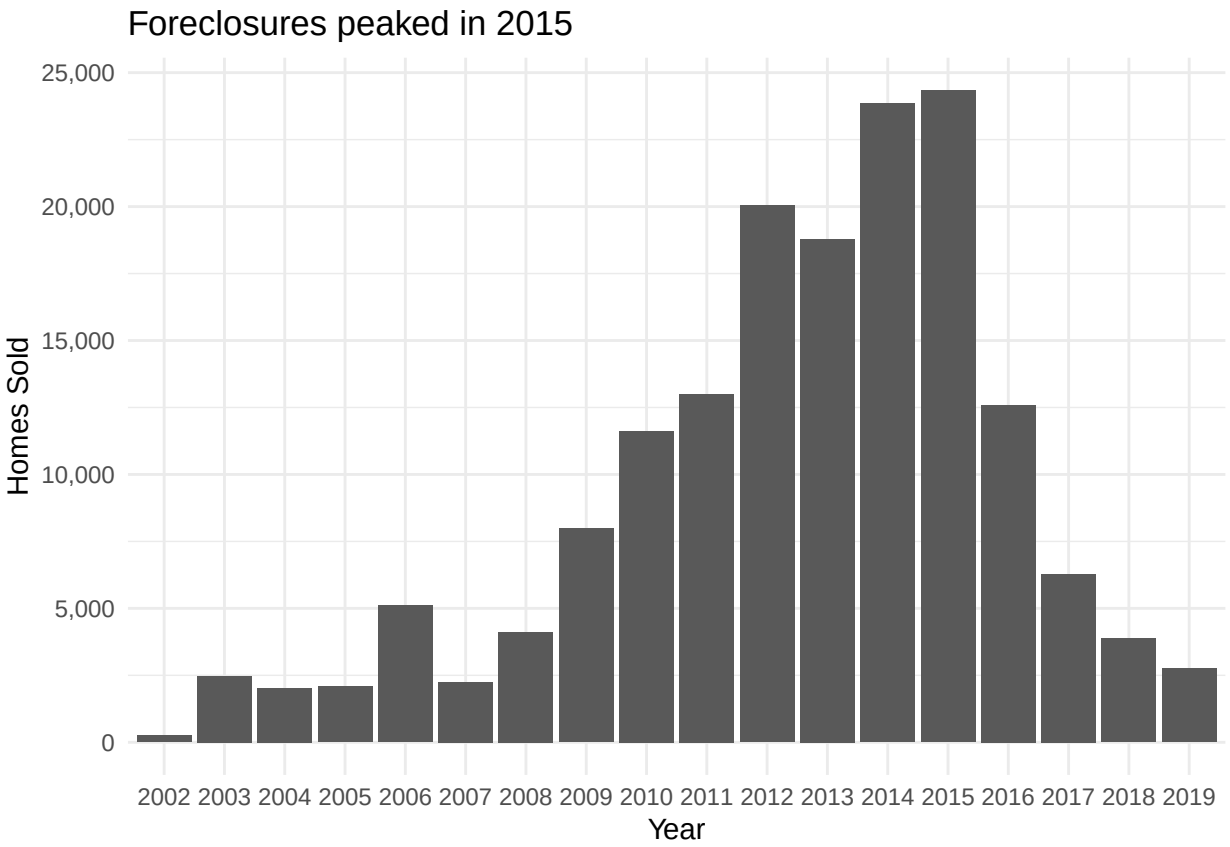
```
properties <- assessments %>%
  count(year) %>%
  filter(year < as_date("2022-01-01"))

homes <- assessments %>%
  filter(propclass == 401) %>%
  count(year) %>%
  filter(year < as_date("2022-01-01"))

ggplot(data = NULL, aes(x = year, y = n)) +
  geom_line(data = properties, color = "black", size = 2, alpha = .5) +
  geom_line(data = homes, color = "blue", size = 2, alpha = .5) +
  scale_x_date(date_labels = "%Y", date_breaks = "2 year") +
  scale_y_continuous(labels = scales::comma,
                     limits = c(0, 355000)) +
  labs(title = "While the number of properties in Detroit has increased\nthe number of homes (blue) has
       x = "Year", y = "Properties") +
  theme_minimal()
```



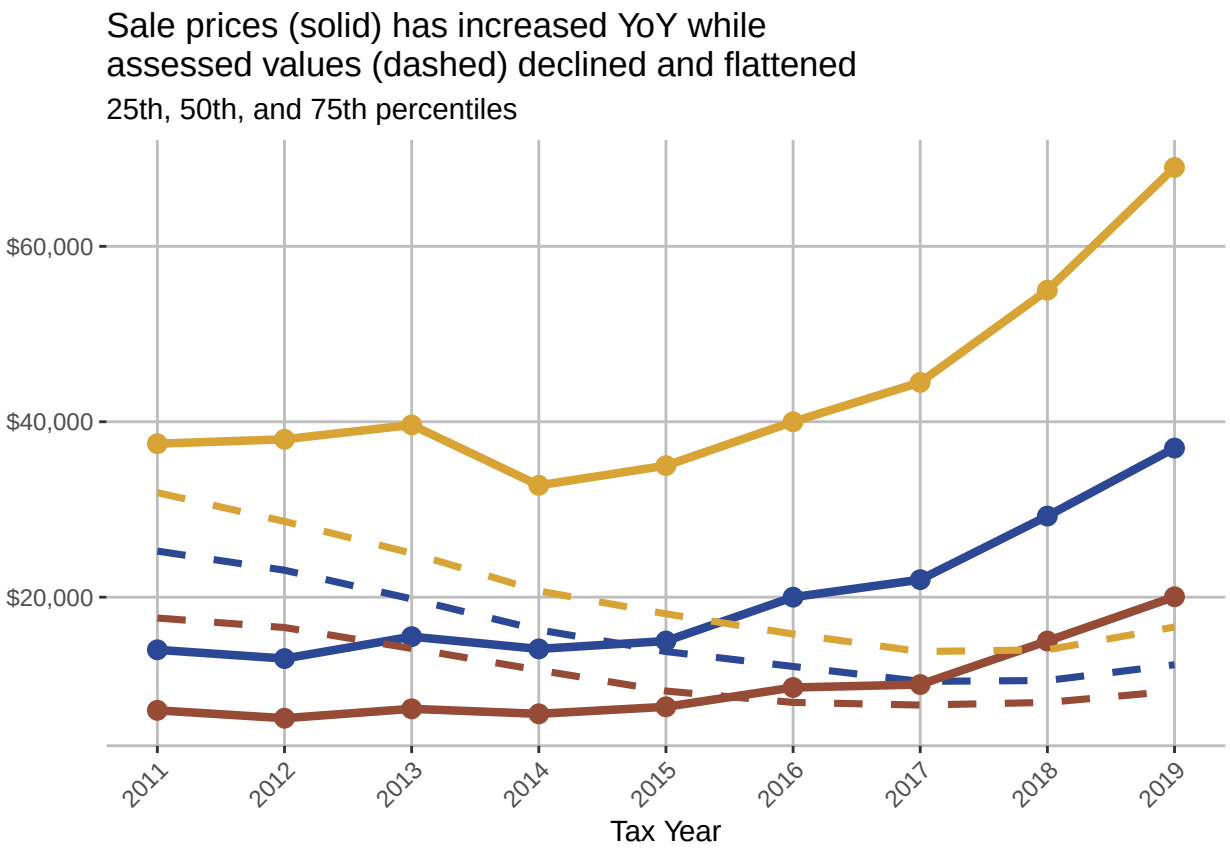

```
parcels %>%
  mutate(sale_year = year(sale_date)) %>%
  select(sale_year, sale_price, assessed_value, property_class) %>%
  filter(sale_price > 4000, assessed_value > 2000, sale_year >= 1990, sale_year <= 2020, property_class
  ggplot(aes(x = sale_year)) +
  geom_bar() +
  scale_y_continuous(labels = scales::comma) +
  labs(x = "Year", y = "Homes Sold",
       title = "Home sales have increased in volume since 2005") +
  theme_minimal()
```




```
foreclosures %>%
  pivot_longer(
    cols = c(`2002`:`2019`),
    names_to = "year"
  ) %>%
  group_by(year) %>%
  summarize(foreclosures = sum(value, na.rm = T)) %>%
  ggplot(aes(x = year, y = foreclosures)) +
  geom_col() +
  scale_y_continuous(labels = scales::comma) +
  labs(x = "Year", y = "Homes Sold",
       title = "Foreclosures peaked in 2015") +
  theme_minimal()
```




```
output[[2]] +
  labs(title = "Sale prices (solid) has increased YoY while\nassessed values (dashed) declined and flat
```



Predicting Overassessments

First we must define an over-assessment. According to this report the legal limit is 50%. We'll use this as the metric. Unfortunately I just don't have time to go in-depth on this homework or I would spend more time on pre-modeling EDA to better determine which characteristics were most important to include. For now I"m going with the "throw spaghetti at the wall" approach and adding as many as I think are initially relevant.

```
model_data <- assessments %>%
  left_join(
    sales %>%
      mutate(sale_year = as_date(sale_year)),
    by = c("PARCELNO" = "parcel_num")
)

model_data <- model_data %>%
  filter(year == "2016-01-01", year(sale_date) == 2016, property_c == "401",
         sale_price > 2500, ASSESSEDVALUE > 1250) %>%
  mutate(over = ifelse(ASSESSEDVALUE * 2 > sale_price, "Over", "Not") %>%
           as.factor())

model_data <- model_data %>%
  left_join(
    parcels %>%
      select(parcel_number, ward, zip_code, use_code_desc, total_square_footage, is_improved, style, ye
```

```
    by = c("PARCELNO" = "parcel_number")
  ) %>%
  mutate(is_improved = ifelse(is_improved == 1, T, F)) %>%
  filter(complete.cases(.)) # Removes ~700 rows
```

For this classification problem I will be using a random forest model. The recipe is going to convert NA values to unknown for zip code, style, and zoning values. Other than that I'm not going to mess with too much pre-processing on my first go.

```
split <- initial_split(model_data)
train <- training(split)
test  <- testing(split)

rf_mod <-
  rand_forest(trees = 1000) %>%
  set_engine("ranger") %>%
  set_mode("classification")

rf_recipe <- recipe(over ~ sale_price + zip_code + is_improved + style + zoning,
                    data = train) %>%
  step_novel(all_nominal_predictors()) %>%
  step_unknown(all_nominal_predictors())

rf_workflow <- workflow() %>%
  add_model(rf_mod) %>%
  add_recipe(rf_recipe)

rf_fit <- fit(rf_workflow, train)

rf_preds <- augment(rf_fit, train)
```

Let's create a classification matrix for this output using the `performanceEstimation` package that I found. fpr stands for false positive, fnr false negative, ppv predictive positive value, etc.

```
performanceEstimation::classificationMetrics(rf_preds$over, rf_preds$.pred_class, metrics = c("fpr", "fn
```

```
##        fpr        fnr        tpr        tnr        ppv
## 0.06358498 0.17687334 0.82312666 0.93641502 0.87293263
```

Not bad!

```
rf_test <- augment(rf_fit, test)

rf_test %>%
  select(PARCELNO, over, .pred_class) %>%
  count(over, .pred_class) %>%
  mutate(pct = n / sum(n))
```

```
## # A tibble: 4 x 4
##   over  .pred_class     n    pct
##   <fct> <fct>       <int>  <dbl>
```

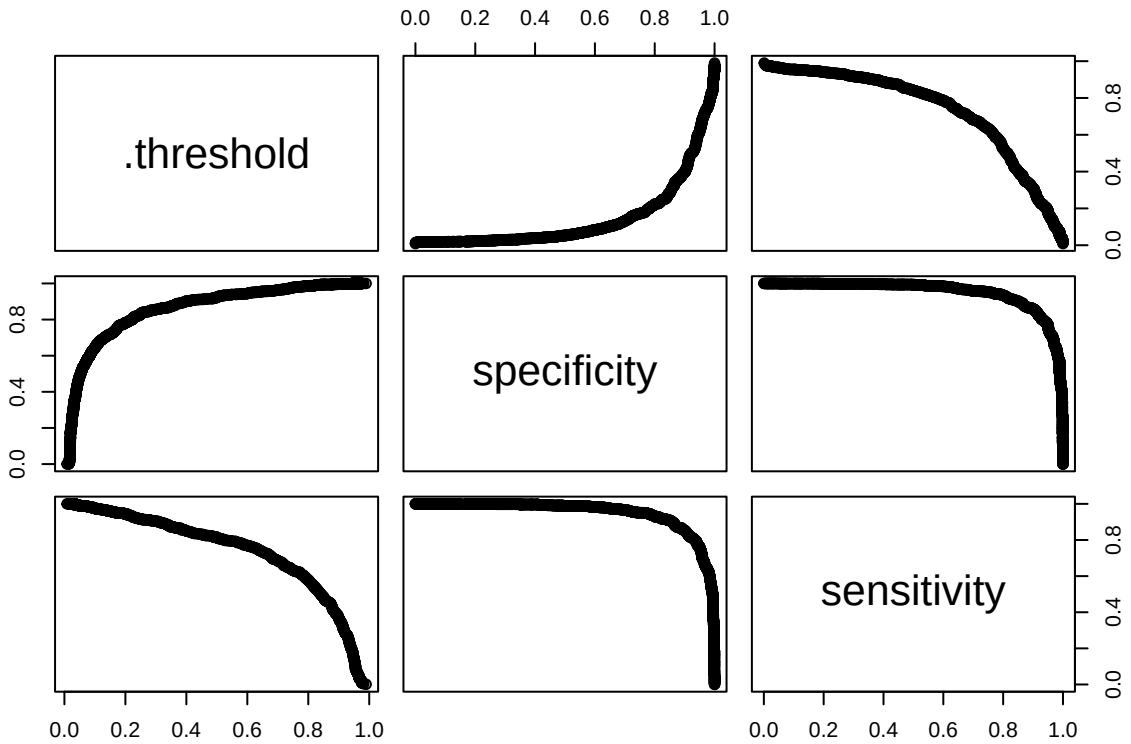
```
## 1 Not   Not           700 0.277
## 2 Not   Over          159 0.0629
## 3 Over  Not           128 0.0506
## 4 Over  Over         1541 0.610
```

```
performanceEstimation::classificationMetrics(rf_test$over, rf_test$.pred_class, metrics = c("fpr", "fnr
```

```
##        fpr        fnr        tpr        tnr        ppv
## 0.07669263 0.18509895 0.81490105 0.92330737 0.84541063
```

Still not bad, that's pretty cool!

```
roc <- yardstick::roc_curve(data = rf_test, truth = over, .pred_Not)

plot(roc)
```



Predicting Assessed Values

This model needs more data than just 2019 to avoid overfitting. I'll add in the previous 4 years. Otherwise this process is exactly the same.

```
model_data <- assessments %>%
  left_join(
```

```
    sales %>%
      mutate(sale_year = as_date(sale_year)),
    by = c("PARCELNO" = "parcel_num")
)

model_data <- model_data %>%
  filter(year(year) <= 2019, year(year) >= 2016,
         year(sale_date) %in% c(2016:2019), property_c == "401",
         sale_price > 2500, ASSESSEDVALUE > 1250) %>%
  mutate(over = ifelse(ASSESSEDVALUE * 2 > sale_price, "Over", "Not") %>%
           as.factor())

model_data <- model_data %>%
  left_join(
    parcels %>%
      select(parcel_number, ward, zip_code, use_code_desc, total_square_footage, is_improved, style, yea
    by = c("PARCELNO" = "parcel_number")
  ) %>%
  mutate(is_improved = ifelse(is_improved == 1, T, F)) %>%
  filter(complete.cases(.)) # Removes ~700 rows
```

Same M.O. as the previous model. I'm going to make minimal changes to the data via recipe.

```
split <- initial_split(model_data)
train <- training(split)
test  <- testing(split)

lm_mod <- linear_reg() %>%
  set_engine("lm") %>%
  set_mode("regression")

lm_recipe <- recipe(ASSESSEDVALUE ~ sale_price + total_square_footage +
                      year_built + is_improved + style,
                    data = train) %>%
  step_novel(all_nominal_predictors()) %>%
  step_unknown(all_nominal_predictors())

lm_workflow <- workflow() %>%
  add_model(lm_mod) %>%
  add_recipe(lm_recipe)

lm_fit <- fit(lm_workflow, train)

lm_preds <- augment(lm_fit, train %>% filter(year == "2019-01-01"))
```

```
bind_rows(
  mape(lm_preds, truth = sale_price, estimate = .pred),
  rmse(lm_preds, truth = sale_price, estimate = .pred)
)
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
```

```
##   <chr>   <chr>          <dbl>
## 1 mape    standard        72.5
## 2 rmse    standard     92127.
```

An average of 72% off across all predictions, and an RMSE of $95,000. How does this predictive power compare to data it hasn't seen?

```
lm_preds <- augment(lm_fit, test %>% filter(year == "2019-01-01"))

bind_rows(
  mape(lm_preds, truth = sale_price, estimate = .pred),
  rmse(lm_preds, truth = sale_price, estimate = .pred)
)
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 mape    standard        72.5
## 2 rmse    standard     84969.
```

A little better, but it is a smaller sample. In the future I could try an approach that weights the data by recency, since we know the average trend in the housing market is "line go up".