House Price Prediction using Locally Weighted Linear Regression

Girish Palya

Contents

TL;DR

Does the value of house depend on the neighborhood it is located in? Real estate agents would say yes. To explore the effect of historical selling prices of houses in the neighborhood on the selling price of a house, locally (geo-spatially) weighted linear regression (LWR) model is used to predict selling prices, and results are compared to prices predicted by parametric linear regression model. LWR makes more accurate predictions compared to linear regression.

Methodology

Parametric learning algorithms base their prediction on calculating a fixed set of parameters (for prediction/hypothesis function) using training data. After the Parameters have been calculated, training data is of no direct consequence during prediction since the same prediction function is used on all test examples.

On the other hand, non-parametric algorithms do not rely on fixed set of parameters. They may calculate parameters for each test observation separately.

Locally weighted linear regression (LWR) is a popular non-parametric learning algorithm, which seeks to minimize following cost function.

$$\sum_{i=1}^{m} w^{(i)}(y^{(i)} - \theta^T X^{(i)})^2$$

Where,

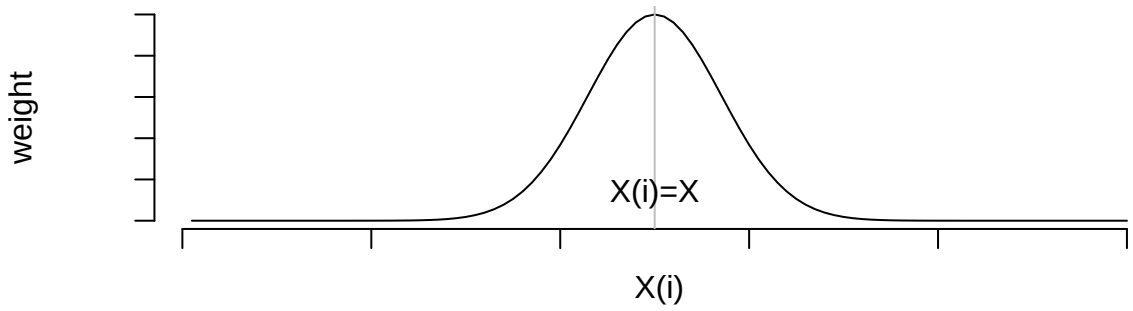
- m: number of training observations
- $w^{(i)}$: weight of the i^{th} observation
- $y^{(i)}$: target/output variable (scalar)
- θ: parameter to be estimated (vector)
- $X^{(i)}$: features/input vector

Weight function is given by

$$w^{(i)} = \exp\left(-\frac{\left(X^{(i)} - X\right)^2}{2\tau^2}\right)$$

If $|X^{(i)} - X|$ is small, $w^{(i)} \approx 1$, and if $|X^{(i)} - X|$ is large, $w^{(i)} \approx 0$. X is the feature vector of test example, and τ is the bandwidth (window size).

Net effect is that cost function sums over only $X^{(i)}$'s that are closer to X, and ignores far away $X^{(i)}$'s. Shape of $w^{(i)}$ is a bell curve (but unlike Gaussian distribution, area under $w^{(i)}$ does not integrate to 1).



Width of the bell curve is determined by the value of τ. Smaller values τ results in narrower peak of the bell curve. When value of τ is small, $X^{(i)}$'s closet to test example X have higher influence on the prediction function. This may result in "overfitting". Conversely, large values of *tau* results in wider window where $X^{(i)}$'s matter for prediction. For sufficiently large values of τ, LWR reduces to parametric linear regression.

Value of τ is approximated through trial-and-error by minimizing mean squared error (MSE) using K-Fold cross validation (k=5). In the regression setting, MSE is the most commonly-used measure.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left(Y^{(i)} - h(X^{(i)})\right)^2$$

where $h(X^{(i)})$ is the prediction that h gives for the ith observation. For linear regression (without weights) h would simply be $\theta^T X^{(i)}$. The MSE will be small if the predicted responses are very close to the true responses, and will be large if for some of the observations, the predicted and true responses differ substantially. Since the MSE is computed using the training data that was used to fit the model, it is really *training* MSE. But we are *interested in knowing how well the model behaves on previously unseen data.* Unfortunately, the dataset only has training data, and thus we limit our analysis to *training* MSE.

LWR has another limitation. If input vector X of test example falls outside the range of training $X^{(i)}$'s, results may not be good. In fact, many learning algorithms are not good at extrapolation.

Dataset

Dataset is available at UCI. Historical market data of real estate valuation is collected from Xindian district of New Taipei City, Taiwan, during a 10 month period in 2012 and 2013.

There are 6 input variables, and 414 observations.

- X1 : Transaction date (for example, 2013.250=2013 March, 2013.500=2013 June, etc.; First digit after the dot is the month)
- X2 : House age (unit: year)
- X3 : Distance to the nearest MRT station (unit: meter)
- X4 : Number of convenience stores in the living circle on foot (unit: integer)
- X5 : Geographic coordinate, latitude (unit: degree)
- X6 : Geographic coordinate, longitude (unit: degree)

Output variable is house price per unit area.

- Y : house price per unit area (10000 New Taiwan Dollar/Ping, where Ping is a local unit. 1 Ping = 3.3 meter squared)

```
library(data.table)
library(rio)
library(caret)
library(geosphere)
library(ggplot2)
require(gridExtra)
url <- paste0("https://archive.ics.uci.edu/ml/machine-learning-databases/00477",
              "/Real%20estate%20valuation%20data%20set.xlsx")
data <- setDT(rio::import(url))
setnames(data, new = sapply(names(data), function(x) gsub(" ", "_", x)))
str(data, vec.len=2.5)
```

```
## Classes 'data.table' and 'data.frame':   414 obs. of  8 variables:
##  $ No                                    : num  1 2 3 4 5 6 ...
##  $ X1_transaction_date                   : num  2013 2013 2014 ...
##  $ X2_house_age                          : num  32 19.5 13.3 13.3 5 7.1 ...
##  $ X3_distance_to_the_nearest_MRT_station: num  84.9 306.6 562 ...
##  $ X4_number_of_convenience_stores       : num  10 9 5 5 5 3 ...
##  $ X5_latitude                           : num  25 25 25 ...
##  $ X6_longitude                          : num  122 122 122 ...
##  $ Y_house_price_of_unit_area            : num  37.9 42.2 47.3 54.8 43.1 32.1 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Data Preparation and EDA

Transaction date (X1) is a time series. All transactions happened in years 2012 and 2013, during which time Taiwan's housing market was in a secular uptrend. Specifically, prices increased by 2.76%, 2.27%, 0.18%, and 2.55% during Q1-Q4 of 2012, and 5.33%, 6.05%, -0.42%, and 3.13% during Q1-Q4 of 2013. Time series is not stationary, and therefore cannot be included directly in regression. Making price prediction at any other time would not be valid. To isolate the effect of other variables, the output variable (house price per unit area) is scaled to adjust for the secular trend. Observations were first recorded in Q3 of 2012. Output variable is scaled from Q4 of 2012 onward, based on the rate of increase of prices in housing market in each quarter.

```
adjustment <- function(month, year) {
  rate_2012 <- c(2.76,  2.27,   0.18,   2.55)
```

```
  rate_2013 <- c(5.33,  6.05,   -0.42, 3.13)
  qtr <- ((month - 1) %/% 3) + 1
  if (year == 2012 & qtr == 3) {
    return(1)  # base rate for Q3 2012
  }
  basis <- 1 + rate_2012[4] / 100
  if (year == 2012) {
    stopifnot(qtr == 4)
    return(basis)
  }
  for (i in 1:qtr) {
    basis <- basis * (1 + rate_2013[qtr] / 100)
  }
  return(basis)
}
data[, year := floor(X1_transaction_date)
     ][, month_num := sapply(X1_transaction_date, function(x) {
       year <- floor(x)
       month_idx <- x %% year
       floor(month_idx * 10) + 1
     })]
data[, Adjusted_house_price_per_unit_area
     := mapply(function(price, m, y) {
       price * adjustment(m, y)
     }, Y_house_price_of_unit_area, month_num, year)]
```

Geographic coordinates of houses are provided as latitude and longitudes. A bounding box of locations can help us tune the window for weighted linear regression. Calculate the geo-spatial (halversine) distance between farthest houses (in meters).

North-south distance:

```
distm(c(0, data[, min(X5_latitude)]),
      c(0, data[, max(X5_latitude)]), fun = distHaversine)
```

```
##          [,1]
## [1,] 9186.084
```

East-west distance:

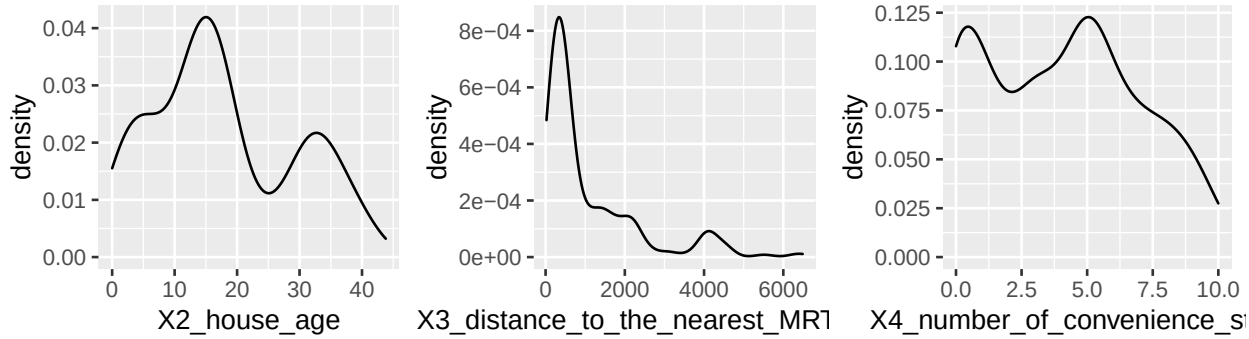
```
distm(c(data[, min(X6_longitude)], 0),
      c(data[, max(X6_longitude)], 0), fun = distHaversine)
```

```
##          [,1]
## [1,] 10323.77
```

Houses are located within 11 kilometer diameter.

Density plot reveals that variables are more-or-less normally distributed. **House age** has anamoly around 25 years. A vast number of houses sold are close to public transport, as indicated by density of **Distance to nearest MRT**. Additionally, majority of houses sold are foot distance away from 0-7 convenience stores.

```
grid.arrange(ggplot(data, aes(x = X2_house_age)) + geom_density(),
             ggplot(data, aes(x = X3_distance_to_the_nearest_MRT_station))
             + geom_density(),
             ggplot(data, aes(x = X4_number_of_convenience_stores))
             + geom_density(), ncol = 3)
```



Correlation between input variables is not significant.

```
correlation_matrix <- cor(data[, 3:5])
prmatrix(correlation_matrix, collab = c("X2", "X3", "X4"))
```

```
##                                             X2          X3          X4
## X2_house_age                         1.00000000  0.02562205  0.04959251
## X3_distance_to_the_nearest_MRT_station 0.02562205  1.00000000 -0.60251914
## X4_number_of_convenience_stores        0.04959251 -0.60251914  1.00000000
```

Results

LWR can give better prediction results compared to linear regression as measured by lower MSE. Weight window parameter τ is kept at 1000. Since distance is measured in meters, this roughly equates to giving more weight to houses within half a kilometer radius (recall that houses in the training set are located within 11 kilometer diameter). LWR appears to be a better choice for a learning algorithm for predicting real estate prices.

```
# Locally weighted linear regression (LWR)
k <- 5
k_folds <- createFolds(data$No, k)
tau <- 1000
MSE <- sapply(names(k_folds), function(fold) {
  train <- data[!No %in% k_folds[[fold]]]
  test <- data[No %in% k_folds[[fold]]]
  predictions <- sapply(seq.int(nrow(test)), function(observation) {
    X <- c(test[observation, X6_longitude], test[observation, X5_latitude])
    halversine <- sapply(seq.int(nrow(train)), function(obs) {
      X_i <- c(train[obs, X6_longitude], train[obs, X5_latitude])
      distm(X_i, X, fun = distHaversine)[1, 1]
    })
    weights <- exp(-(halversine^2 / (2 * tau^2)))
    model <- lm(Adjusted_house_price_per_unit_area
                ~ X2_house_age
```

```
                + X3_distance_to_the_nearest_MRT_station
                + X4_number_of_convenience_stores,
                data = train,
                weights = weights)
    predict(model, test[observation])
  })
  MSE <- sum((test$Adjusted_house_price_per_unit_area - predictions)^2)
  MSE / nrow(test)
})
LWR <- mean(MSE)

# Linear regression
MSE <- sapply(names(k_folds), function(fold) {
  train <- data[!No %in% k_folds[[fold]]]
  test <- data[No %in% k_folds[[fold]]]
  model <- lm(Adjusted_house_price_per_unit_area
                ~ X2_house_age
                + X3_distance_to_the_nearest_MRT_station
                + X4_number_of_convenience_stores,
                data = train)
  predictions <- sapply(seq.int(nrow(test)), function(observation) {
    predict(model, test[observation])
  })
  MSE <- sum((test$Adjusted_house_price_per_unit_area - predictions)^2)
  MSE / nrow(test)
})
LM <- mean(MSE)

# Display results
data.table(Method = c("LWR", "Linear regression"), MSE = c(LWR, LM))
```

```
##               Method       MSE
## 1:               LWR  93.67274
## 2: Linear regression 120.06686
```