Regressions

MKT 566

Instructor: Davide Proserpio

A few things

- Homework 1 and the next homework
- Groups
  - Section 16546 (1 student w/o group)
  - Section 16547 (2 students w/o group)
- Guest speakers:
  - Jonathan Elliot, Director, Data Science at StubHub (Sept. 29)
  - Yang Wang, Principal Economist at Amazon (Nov. 5)
  - Giovanni Marano: Analytics Senior Director at FanDuel (Nov. 17/19)

What we will learn

- We continue to talk about **covariation** and learn how to model it using regressions

- We are going to cover linear regressions and important concepts associated with them

- Chapter 3.4 of R for Marketing Students
- (Advanced & optional) Lecture 6 of Data Storytelling for Marketers

What is a Linear Regression

- In simple terms, a regression allows us to predict a variable Y using one or a set of variables X_j $(j = 1{:}N)$
- We refer to **Y as outcome or dependent variable**
- We refer to $\boldsymbol{X_j}$ **as predictors or independent variables**

- For example:
  - Income (Y) as a function of education (X)
  - Sales (Y) as a function of ad spend (X)
  - Revenue (Y) as a function of review ratings (X)
  - House prices (Y) as a function of mortgage interest rates (X)

What is a Linear Regression

$$Y = F(X)$$

- Where Y is some function of X, i.e., Y depends on X in some way.

- A linear regression simply assumes that the relationship between X and Y is linear

- Machine learning is just building methods to better approximate F(X)

Basic setup and quantities of interest

$$y_i = \beta_o + \beta_1 X_i + \epsilon_i$$

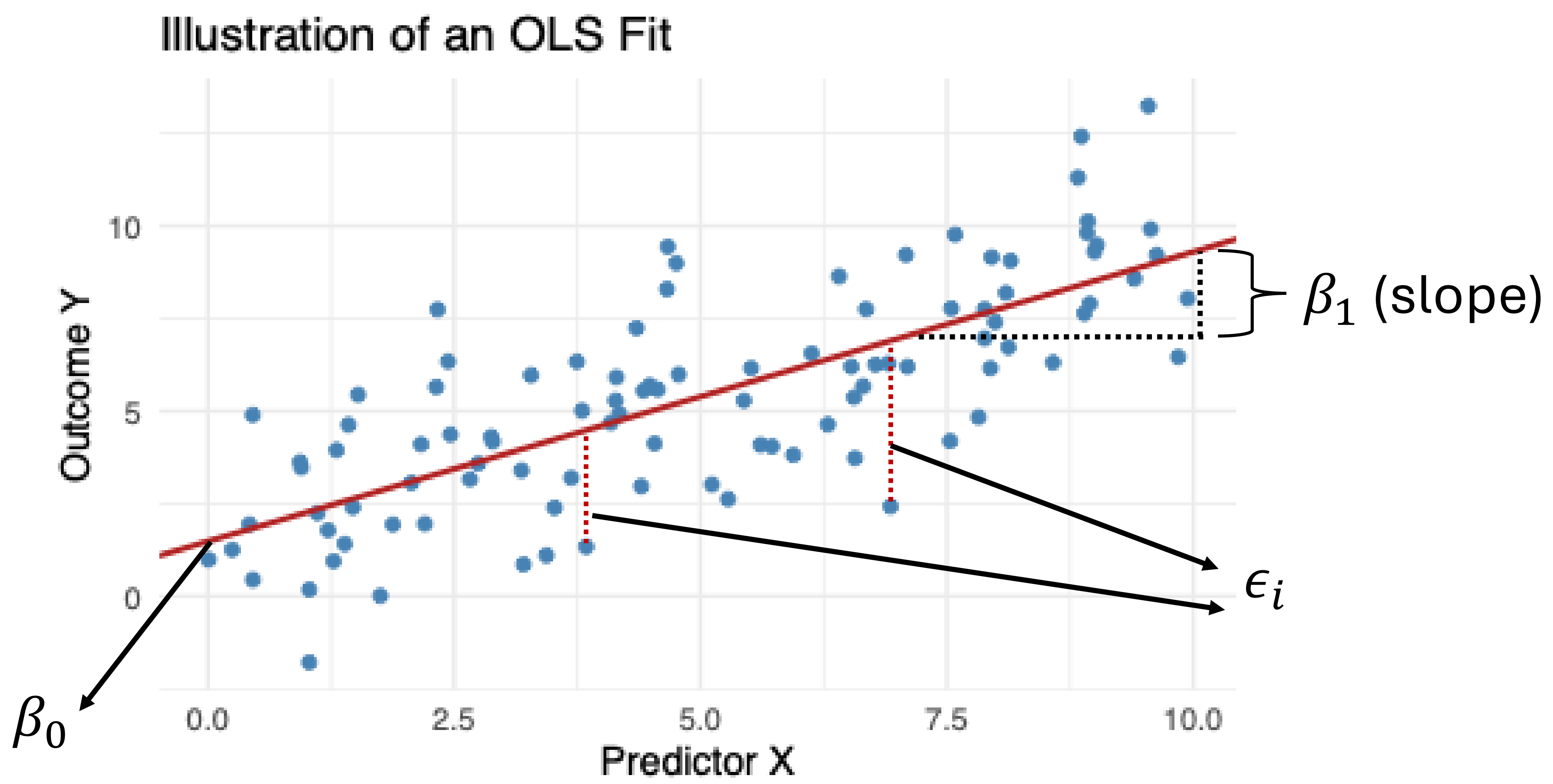
- **X** is the **independent** variable.
- **Y** is the **dependent** variable.
- $\boldsymbol{\beta_0}$ Is the **intercept**.
- $\boldsymbol{\beta_1}$ Is the **coefficient** for variable X.
- $\boldsymbol{\epsilon_i}$ Is the **error term.**

Data

Y	X
3	1
2	5
…	…
…	…
2	4

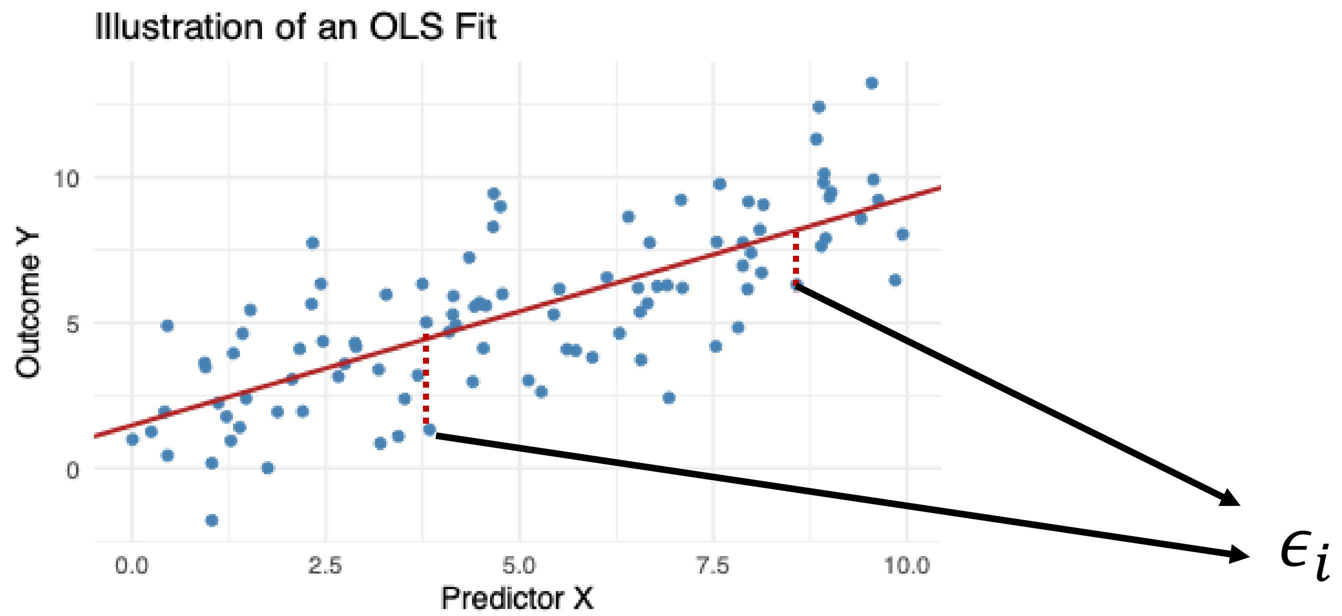
i = 1:N are the rows of the data

Basic setup and quantities of interest



Estimating the coefficient

Ordinary least squares (OLS) estimates the beta coefficients that produce the lowest sum of squared differences between actual and predicted values of the dependent variable



What is Hypothesis Testing?

Hypothesis testing is a way to use data to decide between two claims:

- **Null hypothesis H0**: the default assumption, such as no effect or no difference
- **Alternative hypothesis H1**: the claim we want to test, such as there is an effect or there is a difference

Steps:

- Collect data and compute a test statistic, such as a t-value
- Calculate a p-value, which tells us how likely our data is if H0 were true
- Make a decision:
  - Small p-value → reject H0 and conclude there is evidence for H1
  - Large p-value → fail to reject H0 and conclude there is not enough evidence

Hypothesis Testing for regressions

- **What we test**
  - H_0 :No effect ($\beta = 0$)
  - H_1 :There is an effect ($\beta \neq 0$)
- **How we test**
  - Compute the p-value
- **Decision rule**
  - If **p-value < 0.05** → reject H_0 (evidence of effect)
  - If **p-value ≥ 0.05** → fail to reject H_0 (no strong evidence)

Regression: Quantities of interest

- Coefficients are estimates, therefore, they come with an error
  - **Standard Error (SE) of coefficient:** "How precisely have I pinned down this slope or intercept?" Smaller → more confidence.
  - From SE we can get the **t-statistics** = β_i / SE_i
  - From t-stat, we can get the **p-value**
    - "If there really is no effect (the null is true), what's the probability I'd see data this unusual (or more) just by random luck?"
    - **Low p-value (e.g., 0.05):** Only 5 in 100 random datasets under "no effect" would look this extreme → so you start to doubt the "no effect" story.
    - Generally speaking, if p-value ≤ 0.05, we say the coefficient is **statistically significant**, i.e., different from zero.

- **Smaller SE → larger t-stat → smaller p-value →** stronger evidence against the null hypothesis

A little more technical summary

- **Coefficients**
  - Estimated effects of predictors on the outcome
  - Always estimates → subject to sampling error
- **Standard Error (SE)**
  - Precision of coefficient estimate
  - Smaller SE ⇒ more confidence in the estimate
- **t-statistic**
  - $t_i = \frac{\widehat{\beta}_i}{SE(\widehat{\beta}_i)}$
  - Measures how many SEs away the coefficient is from zero
- **p-value**
  - Probability of observing a t-stat as extreme as this if the true effect is 0
  - Smaller p-value ⇒ stronger evidence against the null hypothesis

Measure of fit

How do we know if our regression is doing a good job at predicting Y?

R-squared (R^2) is a summary statistic in regressions that tells you how well your model's predictions match the actual data

$$R^2 = \frac{\text{Explained Sum of Squares}}{\text{Total Sum of Squares}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

- $\sum (y_i - \hat{y}_i)^2$ = residual (unexplained) variance
- $\sum (y_i - \bar{y})^2$ = total variance in the outcome

Measure of fit

How do we know if our regression is doing a good job at predicting Y?

R-squared (R^2) is a summary statistic in regressions that tells you how well your model's predictions match the actual data

Model error

$$R^2 = \frac{\text{Explained Sum of Squares}}{\text{Total Sum of Squares}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Error using the mean

- $\sum (y_i - \hat{y}_i)^2$ = residual (unexplained) variance
- $\sum (y_i - \bar{y})^2$ = total variance in the outcome

What does β_1 tell us?

- **All else equal** (ceteris paribus), how much Y changes as a function of X
- The interpretation depends on the regression functional form

	Model Form	Regression Equation	Interpretation of β_1
1.	**Level–Level**	$Y = \beta_0 + \beta_1 X$	A one-unit increase in X ⇒ a β_1-unit change in Y.
2.	**Log–Level**	$\ln Y = \beta_0 + \beta_1 X$	A one-unit increase in X ⇒ an approximately $\beta_1 \times 100\%$ change in Y.
3.	**Level–Log**	$Y = \beta_0 + \beta_1 \ln X$	A 1% increase in X ⇒ an approximately $\frac{\beta_1}{100}$-unit change in Y.
4.	**Log–Log**	$\ln Y = \beta_0 + \beta_1 \ln X$	A 1% increase in X ⇒ an approximately $\beta_1\%$ change in Y.

Note: for model 2-4, these approximations hold for small changes in X and/or β

Why log numeric variables?

- **Linearizes Nonlinear Relationships**
  - Many relationships in economics and social science are **multiplicative or curved**, not straight lines.
  - Example: A $1 increase in price affects demand **very differently** when price goes from:
    - $5 → $6 vs.
    - $100 → $101
  - Taking the log of, say price, **linearizes** this relationship, making it easier for a linear model to fit.

Why log numeric variables?

Reduces Skewness

- Variables like price and income variables are often **right-skewed** (many small values, few large ones).
- Taking the log:
  - Compresses large values
  - Expands small differences among low values → Makes the distribution more symmetric and closer to normal
- This can improve model performance and **make OLS assumptions (like normality of errors)** more realistic.

Why log numeric variables?

Reduces the influence of outliers

- Large numeric variables can **dominate the regression**, especially if they contain outliers.
- Logging reduces their influence, which can help with:
  - Numerical stability
  - More robust coefficient estimates

Why log numeric variables?

Interpretability: Elasticities

- When you use log of, e.g., price, coefficients are easier to interpret:
- In a **log-log model**, the coefficient is an **elasticity**: "A 1% increase in price → X% change in demand"
- In a **log-level model**, the coefficient tells you the **percentage change in the outcome** from a one-unit change in price.
- These interpretations are more intuitive, especially in economics or marketing

Multiple independent variables

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \cdots + \beta_j x_{j,i} + \epsilon_i$$

- Everything I just discussed applies!

Estimating linear models in R

```
# estimate the linear model
model = lm(y ~ x, data = yourdata)
# print the results
summary(model)
```

Library for creating pretty tables: **stargazer**

Example: Airbnb dataset

- A cross-sectional dataset of about 50k Airbnb listings in the U,S with some variables describing the listing
  - Cities: Austin, Boston, Los Angeles, Miami, NYC

```
> head(airbnb)
   listing_id price bathrooms bedrooms cancellation_policy guests_included property_type zipcode star_rating reviews_count        city    room_type
        <int> <int>     <num>    <int>              <char>           <int>        <char>  <char>       <num>         <int>      <char>       <char>
1:        147   238         1        2              strict               4         House   90291         5.0            79 Los Angeles  Entire home
2:       1078    89         1        1            flexible               2     Apartment   78705         5.0           117      Austin  Entire home
3:       2055    89         1        1            moderate               1         House   33145         5.0            91       Miami Private room
4:       2265   175         2        2              strict               2         House   78702         4.5            15      Austin  Entire home
5:       3021   120         1        1              strict               2         House   90046         4.5             4 Los Angeles  Entire home
6:       3319   119         1        1              strict               1     Apartment   90048         5.0           298 Los Angeles  Entire home
```

Example: Airbnb dataset

- Let’s predict price as a function of the number of reviews a listing has
- What do you expect the relationship to be?

Example: Airbnb dataset

Example: Airbnb dataset

```
m1 = lm(price ~ reviews_count, data = airbnb)
summary(m1)
```

```
Call:
lm(formula = price ~ reviews_count, data = airbnb)

Residuals:
   Min     1Q Median     3Q    Max 
-145.0  -74.9  -39.5   23.3 9855.4 

Coefficients:
               Estimate Std. Error t value            Pr(>|t|)    
(Intercept)   148.04066    0.90102  164.30 <0.0000000000000002 ***
reviews_count  -0.34672    0.03195  -10.85 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 165.9 on 50834 degrees of freedom
Multiple R-squared:  0.002311,	Adjusted R-squared:  0.002291 
F-statistic: 117.8 on 1 and 50834 DF,  p-value: < 0.00000000000000022
```

Example: Airbnb dataset

```
m1 = lm(price ~ reviews_count, data = airbnb)
summary(m1)
```

```
Call:
lm(formula = price ~ reviews_count, data = airbnb)

Residuals:
   Min      1Q  Median      3Q     Max 
-145.0   -74.9   -39.5    23.3  9855.4 

Coefficients:
               Estimate Std. Error t value             Pr(>|t|)    
(Intercept)   148.04066    0.90102  164.30 <0.0000000000000002 ***
reviews_count  -0.34672    0.03195  -10.85 <0.0000000000000002 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 165.9 on 50834 degrees of freedom
Multiple R-squared:  0.002311,	Adjusted R-squared:  0.002291 
F-statistic: 117.8 on 1 and 50834 DF,  p-value: < 0.00000000000000022
```

Example: Airbnb dataset

```
m1 = lm(price ~ reviews_count, data = airbnb)
summary(m1)
```

```
Call:
lm(formula = price ~ reviews_count, data = airbnb)

Residuals:
   Min     1Q Median     3Q    Max 
-145.0  -74.9  -39.5   23.3 9855.4 

Coefficients:
               Estimate Std. Error t value            Pr(>|t|)    
(Intercept)   148.04066    0.90102  164.30 <0.0000000000000002 ***
reviews_count  -0.34672    0.03195  -10.85 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 165.9 on 50834 degrees of freedom
Multiple R-squared:  0.002311,	Adjusted R-squared:  0.002291 
F-statistic: 117.8 on 1 and 50834 DF,  p-value: < 0.00000000000000022
```

Understanding Regression Output

- **Residual Standard Error: 165.9**
  - On average, model predictions are about **$166 off** from the actual values
- **Multiple R-squared: 0.0023**
  - The model explains **0.23% of the variation** in the outcome
  - Very low explanatory power
- **Adjusted R-squared: 0.0023**
  - Same as R^2, but penalizes adding useless predictors
- **F-statistic: 117.8, p-value < 2.2e-16**
  - Tests whether the predictor(s) together explain anything at all
  - Large F and tiny p-value mean: **Yes, the predictor matters statistically**

Example: Airbnb dataset

```
library(stargazer)
# estimate the model
m1 = lm(price ~ reviews_count, data = airbnb)
# create a pretty table
stargazer(m1,
      type = "text",
      title = "Regression of Price on Number of Reviews"
      dep.var.labels = "Price",
      covariate.labels = "Number of Reviews",
      omit.stat = c("f", "ser”, “adj.rsq”),
      digits = 2)
```

```
Regression of Price on Number of Reviews
==============================================
                        Dependent variable:
                    --------------------------
                              Price
----------------------------------------------
Number of Reviews           -0.35***
                             (0.03)

Constant                   148.04***
                             (0.90)

----------------------------------------------
Observations                 50,836
R2                           0.002
==============================================
Note:              *p<0.1; **p<0.05; ***p<0.01
```

Including categorical variables

- **How does R deal with categorical variables? Using factors**
  - A **factor** is R's special way of storing **categorical variables** (things like *city*, *gender*, *yes/no*, etc.).
  - Under the hood, a factor is just **numbers with labels**.
  - Example:

```r
city <- factor(c("NYC", "LA", "Miami", "NYC"))
city
# [1] NYC    LA     Miami NYC
# Levels: LA Miami NYC
```

- Here, LA = 1, Miami = 2, NYC = 3 internally.
- R stores numbers, but shows you labels.

Example: Airbnb dataset

- Let's regress price on city
- We have five values:
  - Austin, Boston, Los Angeles, Miami, NYC

```
m1 = lm(price ~ city, data = airbnb)
# Create table
stargazer(m1, type = "text", title =
"Regression of Price on City",
          dep.var.labels = "Price",
          omit.stat = c("f", "ser",
"adj.rsq"), digits = 2)
```

```
Regression of Price on City
=============================================
                      Dependent variable:
                  ---------------------------
                             Price
---------------------------------------------
cityBoston                 -17.60***
                            (2.67)

cityLos Angeles            -36.77***
                            (1.91)

cityMiami                  -40.93***
                            (2.59)

cityNew York City           -28.17
                            (39.03)

Constant                   169.95***
                            (1.63)

---------------------------------------------
Observations                50,836
R2                           0.01
=============================================
Note:             *p<0.1; **p<0.05; ***p<0.01
```

Example: Airbnb dataset

Why do we see only four coefficients?

Obs	**City**	**Austin**	**Boston**	**Los Angeles**	**Miami**
1	Austin	1	0	0	0
2	Boston	0	1	0	0
3	Los Angeles	0	0	1	0
4	Miami	0	0	0	1
5	New York City	0	0	0	0

Example: Airbnb dataset

- Let’s regress price on city
- We have five values:
  - Austin, Boston, Los Angeles, Miami, NYC

```
m1 = lm(price ~ city, data = airbnb)
# Create table
stargazer(m1, type = "text", title =
"Regression of Price on City",
          dep.var.labels = "Price",
          omit.stat = c("f", "ser",
"adj.rsq"), digits = 2)
```

```
Regression of Price on City
=============================================
                      Dependent variable:
                  ---------------------------
                             Price
---------------------------------------------
cityBoston                 -17.60***
                            (2.67)

cityLos Angeles            -36.77***
                            (1.91)

cityMiami                  -40.93***
                            (2.59)

cityNew York City           -28.17
                            (39.03)

Constant                   169.95***
                            (1.63)

---------------------------------------------
Observations                50,836
R2                           0.01
=============================================
Note:              *p<0.1; **p<0.05; ***p<0.01
```

Austin avg. price

Example: Airbnb dataset

Change the base level city:

```
# convert city to factor
airbnb$city = as.factor(airbnb$city)
# set a different level
airbnb$city = relevel(airbnb$city, ref = "New York City")
```

```
=========================================
                    Dependent variable:
                ---------------------------
                          Price
-----------------------------------------
cityAustin                28.17
                         (39.03)

cityBoston                10.57
                         (39.05)

cityLos Angeles           -8.61
                         (39.01)

cityMiami                -12.76
                         (39.05)

Constant                141.78***
                         (38.99)

-----------------------------------------
Observations             50,836
R2                        0.01
=========================================
Note:          *p<0.1; **p<0.05; ***p<0.01
```