Introduction to Regression and Model Fit

Ivan Corneillet

Data Scientist

GENERAL ASSEMBLY

Learning Objectives

After this lesson, you should be able to:

- Define simple linear regression and multiple linear regression
- Build a linear regression model using a dataset that meets the linearity assumption
- Evaluate model fit
- Understand and identify multicollinearity in a multiple regression

Announcements and Exit Tickets

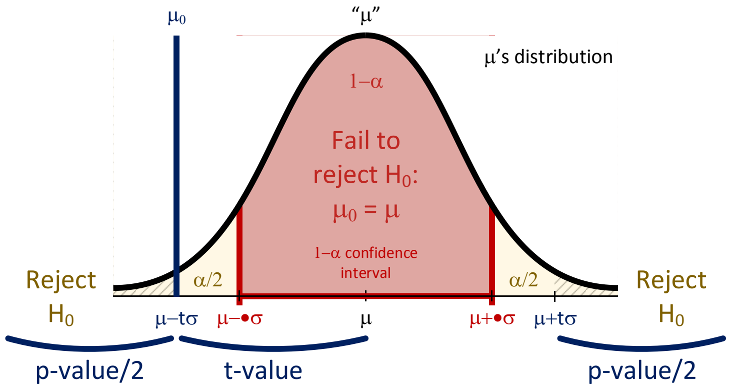
Q & A

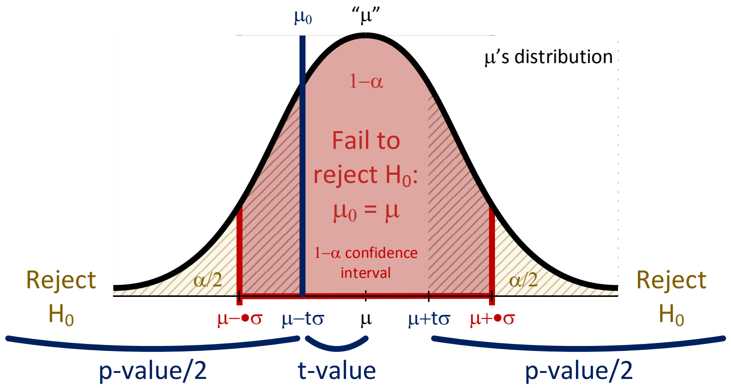
Review

Review

Two-Tail Hypothesis Testing

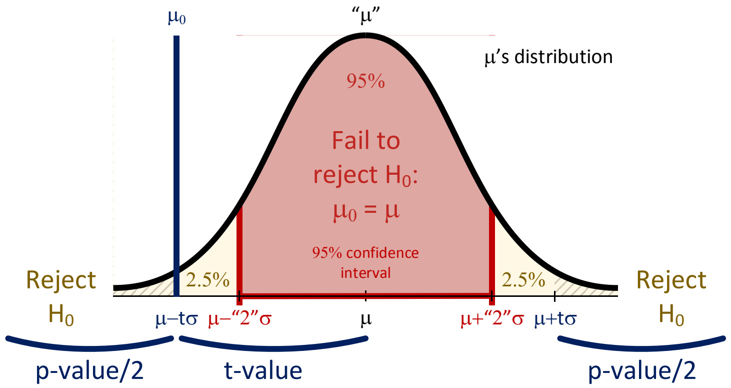
Two-Tail Hypothesis Testing

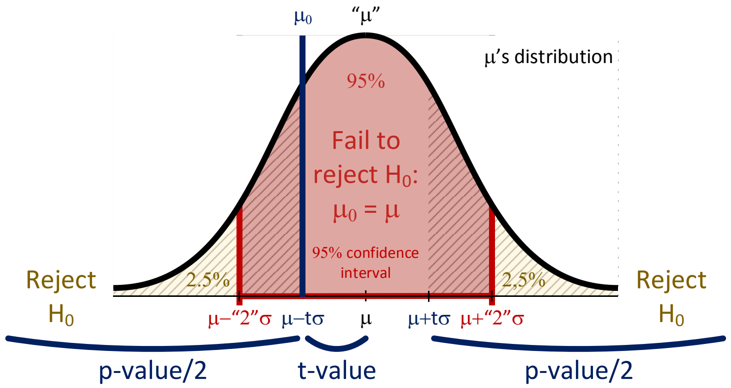




|t-value|	p-value	$1-\alpha$ Confidence Interval $([\mu_0 -\cdot\, \sigma, \mu_0 +\cdot\, \sigma])$	H$_0$ / H$_a$	Conclusion
$\geq \cdot$	$\leq \alpha$	μ_0 is outside	Found evidence that $\mu \neq \mu_0$: Reject H$_0$	$\mu \neq \mu_0$
$< \cdot$	$> \alpha$	μ_0 is inside	Did not find that $\mu \neq \mu_0$: Fail to reject H$_0$	$\mu = \mu_0$

Two-Tail Hypothesis Testing ($\alpha = .05$)





|t-value|	p-value	$1-\alpha$ Confidence Interval $([\mu_0 - 2\sigma, \mu_0 + 2\sigma])$	H_0 / H_a	Conclusion
$\geq$ "~2"(*) (*) (check t-table)	$\leq .025$	μ_0 is outside	Found evidence that $\mu \neq \mu_0$: Reject H_0	$\mu \neq \mu_0$
$<$ "~2"	$> .025$	μ_0 is inside	Did not find that $\mu \neq \mu_0$: Fail to reject H_0	$\mu = \mu_0$

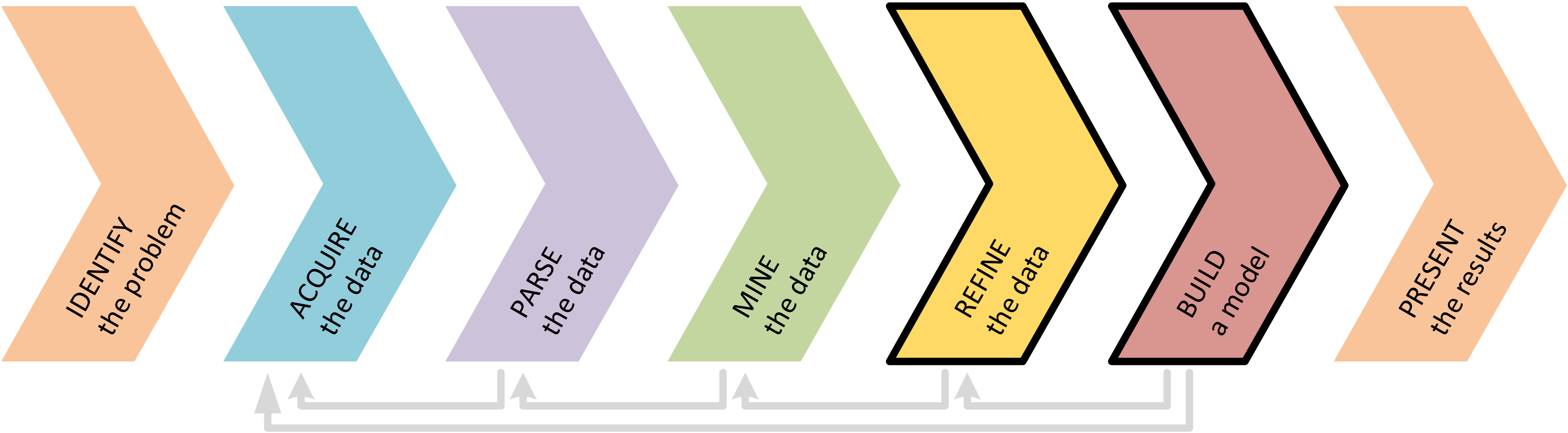
Unit 2

Overview

Unit 2 – Foundation of Modeling

Research Design and Data Analysis	Research Design	Data Visualization in *pandas*	Statistics	Exploratory Data Analysis in *pandas*
Foundations of Modeling	Linear Regression *(sessions 6 and 7)*	Classification Models *(sessions 8 and 9)*	Evaluating Model Fit *(sessions 6 and 7)*	Presenting Insights from Data Models *(sessions 10)*
Data Science in the Real World	Decision Trees and Random Forests	Time Series Data	Natural Language Processing	Databases

Unit 2 and the Data Science Workflow



Unit 2 and the Data Science Workflow (cont.)

❺ Refine the Data

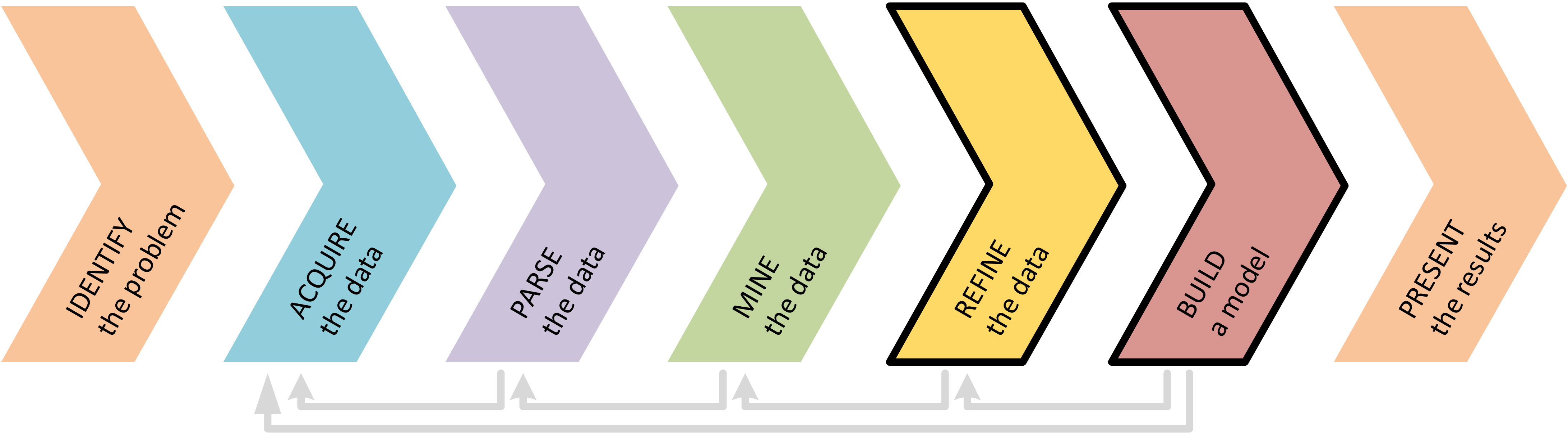
- Identify trends and outliers *(session 3)*
- Apply descriptive *(session 3)* and inferential statistics *(session 5)*
- Document *(session 2)* and **transform data** *(units 2-3)*

❻ Build a Model

- **Select appropriate model** *(units 2-3)*
- **Build model** *(units 2-3)*
- **Evaluate** *(session 5; units 2-3)* and **refine model** *(units 2-3)*

Today

Today we keep our focus on the REFINE the data and BUILD a model steps but with (1) a focus on linear regression modeling and (2) what the inferential statistics tell us about the fit of these linear models



Today (cont.)

Research Design and Data Analysis	Research Design	Data Visualization in *pandas*	Descriptive Statistics for Exploratory Data Analysis Inferential Statistics for Model Fit	Exploratory Data Analysis in *pandas*
Foundations of Modeling	Linear Regression	Classification Models	Evaluating Model Fit	Presenting Insights from Data Models
Data Science in the Real World	Decision Trees and Random Forests	Time Series Data	Natural Language Processing	Databases

Here’s what’s happening today:

- Final Project 1 due today
- Announcements and Exit Tickets
- Review
- Unit 2 Overview
- ❺ Refine the Data and ❻ Build a Model | Simple Linear Regression
  - Variable Transformations
  - How is a regression model fitted to a dataset?
  - Common regression assumptions
  - How to check modeling assumptions
  - How to check normality assumption
  - Inference and Fit and R^2 (r-square)
- ❺ Refine the Data and ❻ Build a Model | Multiple Linear Regression
  - How to interpret the model’s parameters
  - Multicollinearity
  - $\bar{R}^2$ (adjusted R^2)
- Lab – Introduction to Regression and Model Fit
- Review

DS

Simple Linear Regression

Simple Linear Regression

- The simple linear regression model captures a linear relationship between a single input variable x and a response variable y

$$y = \beta_0 + \beta_1 \cdot x + \varepsilon$$

- y is the **response** variable (what we want to predict); also called *dependent* variable or *endogenous* variable
- x is the **explanatory** variable (what we use to train the model); also called *independent* variable, *exogenous* variable, *regressor*, or *feature*
- β_0 and β_1 are the **regression's coefficients**; also called the model's parameters
  - β_0 is the line's intercept; β_1 is the line's slope
- ε is the **error** term; also called the residual

Simple Linear Regression (cont.)

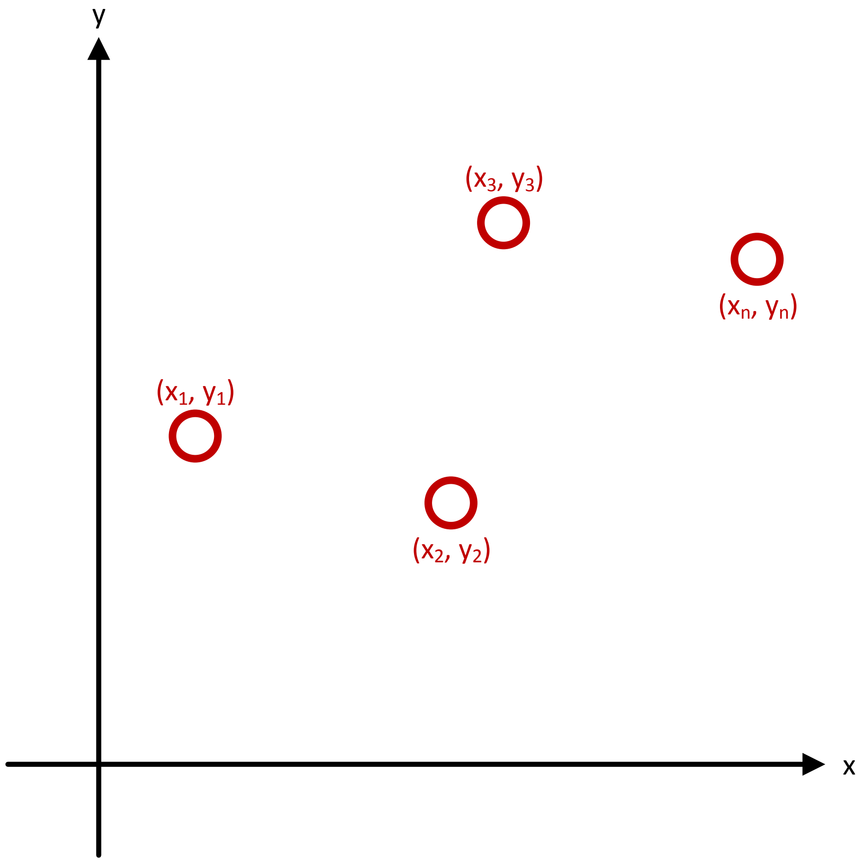
- Given $x = (x_1, x_2, \dots, x_n)$ and $y = (y_1, y_2, \dots, y_n)$, we can formulate the linear model as

$$y_i = \beta_0 + \beta_1 \cdot x_i + \varepsilon_i$$

- In our Python environment, x and y represent *pandas Series* and x_i and y_i their values at row $i - 1$ (remember that indices in Python start at 0…)
- E.g. (SF housing dataset),
  - x is the property's size (`df.Size`)
  - y is the property's sale price (`df.SalePrice`)

Simple Linear Regression (cont.)

- In words, this equation says that for each observation i, y_i can be explained by $\beta_0 + \beta_1 \cdot x_i$
- ε_i is a "white noise" disturbance which we do not observe
  - ε_i models how the observations deviate from the exact slope-intercept relation
- We do not observe the constants β_0 or β_1 either, so we have to estimate them



Simple Linear Regression (cont.)

- Given estimates for the model coefficients $\hat{\beta}_0$ (β_0 hat) and $\hat{\beta}_1$, we predict y using

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \cdot x$$

- The hat symbol (^) denotes an estimated value

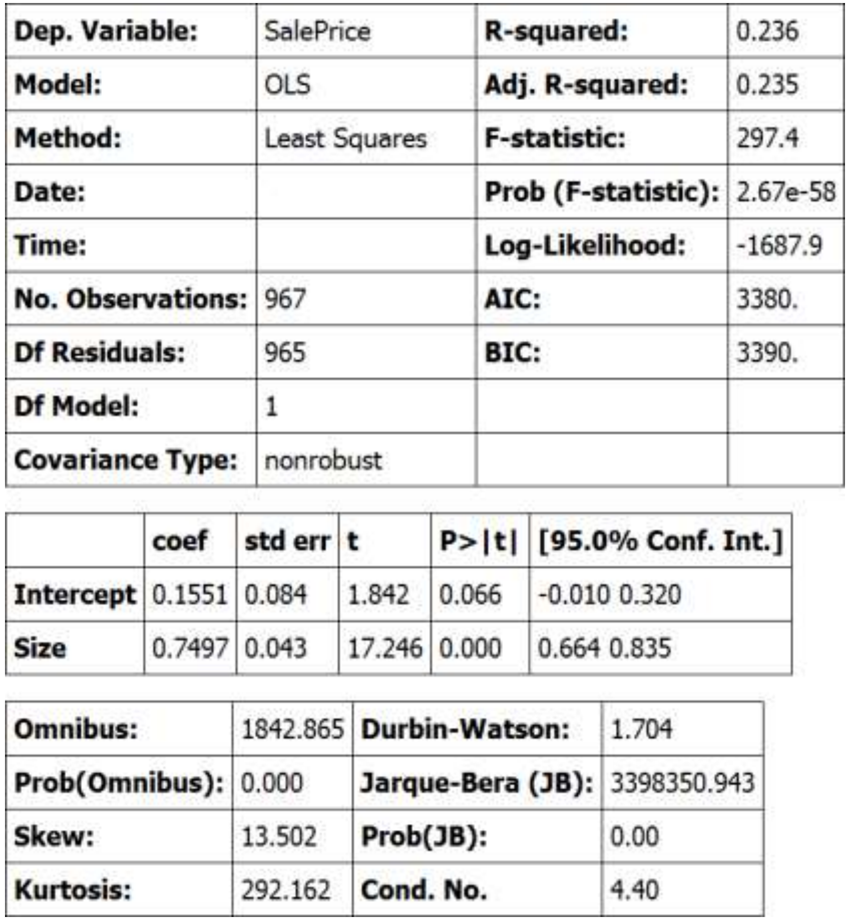
- E.g. (SF housing dataset),

$$\widehat{SalePrice} = \hat{\beta}_0 + \hat{\beta}_1 \cdot Size$$

Simple Linear Regression

Codealong – Part A1
Variable Transformations
Simple Linear Regression

SalePrice ~ Size



Dep. Variable:	SalePrice	R-squared:	0.236
Model:	OLS	Adj. R-squared:	0.235
Method:	Least Squares	F-statistic:	297.4
Date:		Prob (F-statistic):	2.67e-58
Time:		Log-Likelihood:	-1687.9
No. Observations:	967	AIC:	3380.
Df Residuals:	965	BIC:	3390.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	0.1551	0.084	1.842	0.066	-0.010 0.320
Size	0.7497	0.043	17.246	0.000	0.664 0.835

Omnibus:	1842.865	Durbin-Watson:	1.704
Prob(Omnibus):	0.000	Jarque-Bera (JB):	3398350.943
Skew:	13.502	Prob(JB):	0.00
Kurtosis:	292.162	Cond. No.	4.40

$$SalePrice\ [\$M] = \underbrace{.155}_{\widehat{\beta}_0} + \underbrace{.750}_{\widehat{\beta}_1} \times Size\ [1{,}000\ sqft]$$

(the slope is significant but not the intercept)

Simple Linear Regression

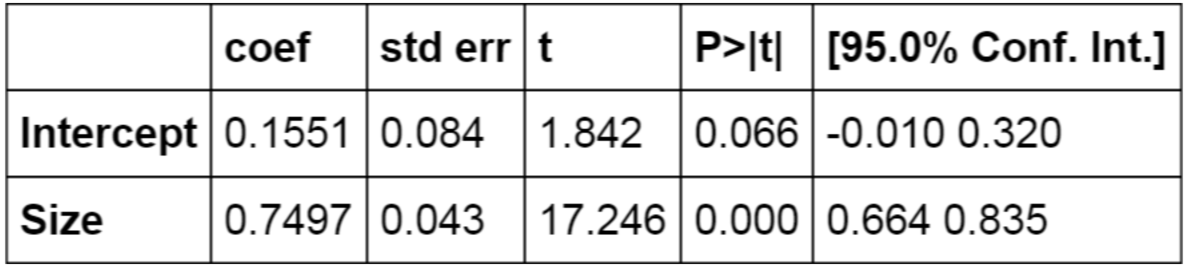
Activity | Knowledge Check

Activity | Knowledge Check

EXERCISE

DIRECTIONS (10 minutes)

1. Using the table below,
   a. How do you interpret the model's parameters? (intercept and slope)
   b. Replace all question marks in the handout with their value or answer



	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	0.1551	0.084	1.842	0.066	-0.010 0.320
Size	0.7497	0.043	17.246	0.000	0.664 0.835

2. When finished, share your answers with your table

DELIVERABLE

Answers to the above questions

Activity | Knowledge Check (cont.)

$Intercept(\beta_0) = .155$

- $Intercept = SalePrice\ [\$M]\ when\ Size = 0$
- $Intercept = \$0.155M = \$155k$
- The simple linear regression predicts that a property of 0 sqft would sell for $155k

$Slope(\beta_1) = .750$

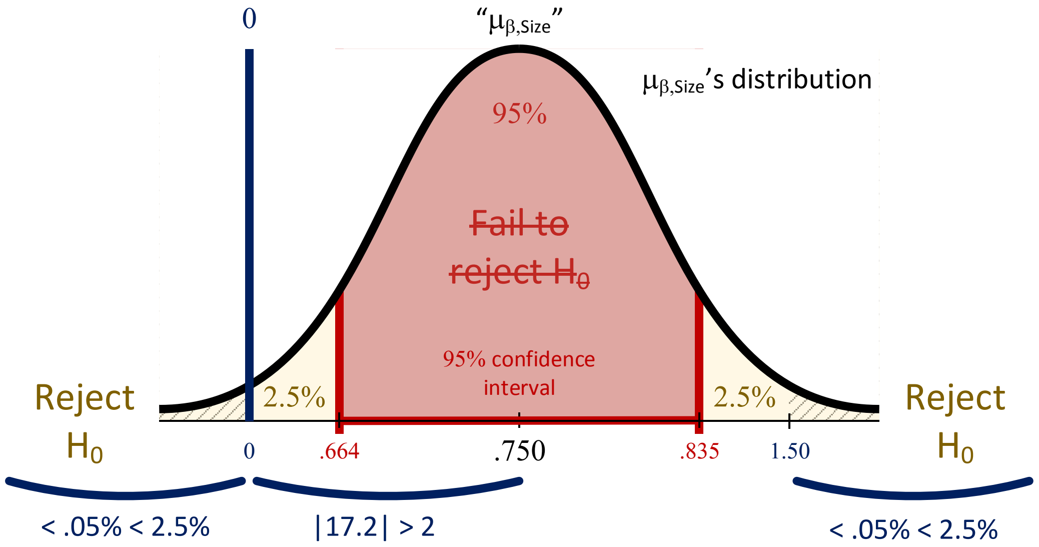
- $Slope = \frac{SalePrice\ [\$M] - Intercept\ [\$M]}{Size[1{,}000\ sqft]}$
- $Slope = .750\ [\$M\ per\ 1{,}000\ sqft] = \$750k/1{,}000\ sqft$
- The simple linear regression predicts that buyers would pay an $750k for each 1,000 sqft

Activity | Knowledge Check (cont.)

Size

Reject $H_0: \mu_{\beta_{Size}} = 0$

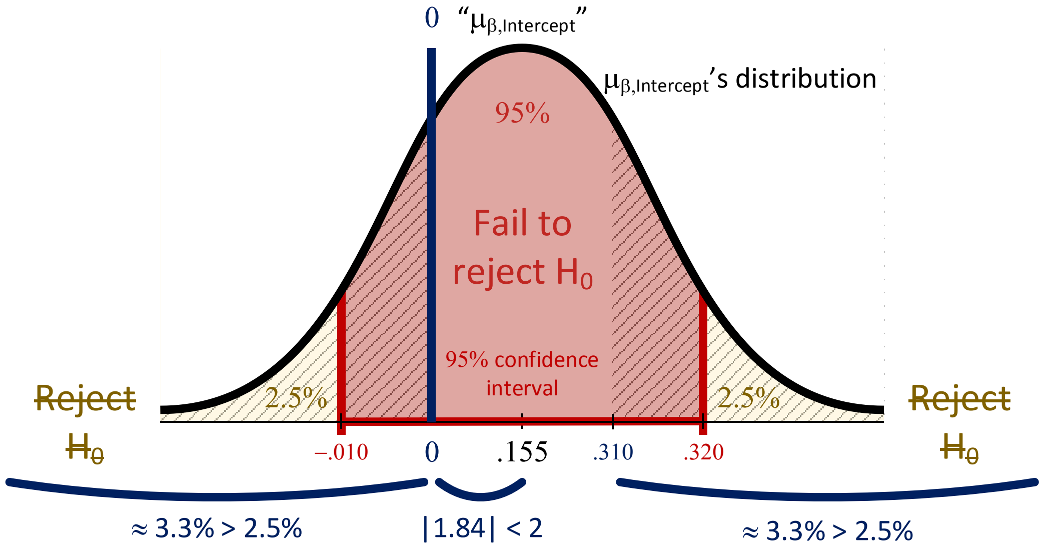
Size is significant



Intercept?

Fail to reject $H_0: \mu_{\beta_{Intercept}} = 0$

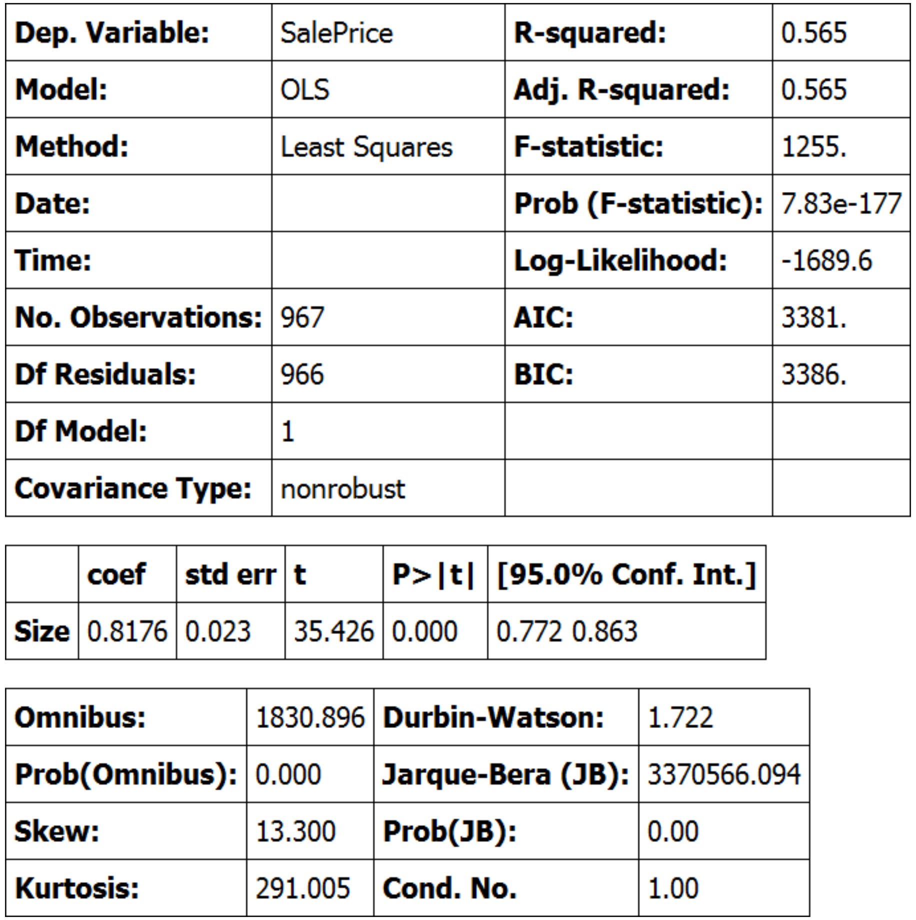
Intercept is not significant?



Simple Linear Regression

Codealong – Part A2
Simple Linear Regression

`SalePrice ~ 0 + Size` (‘0’ meaning the intercept is forced to 0)



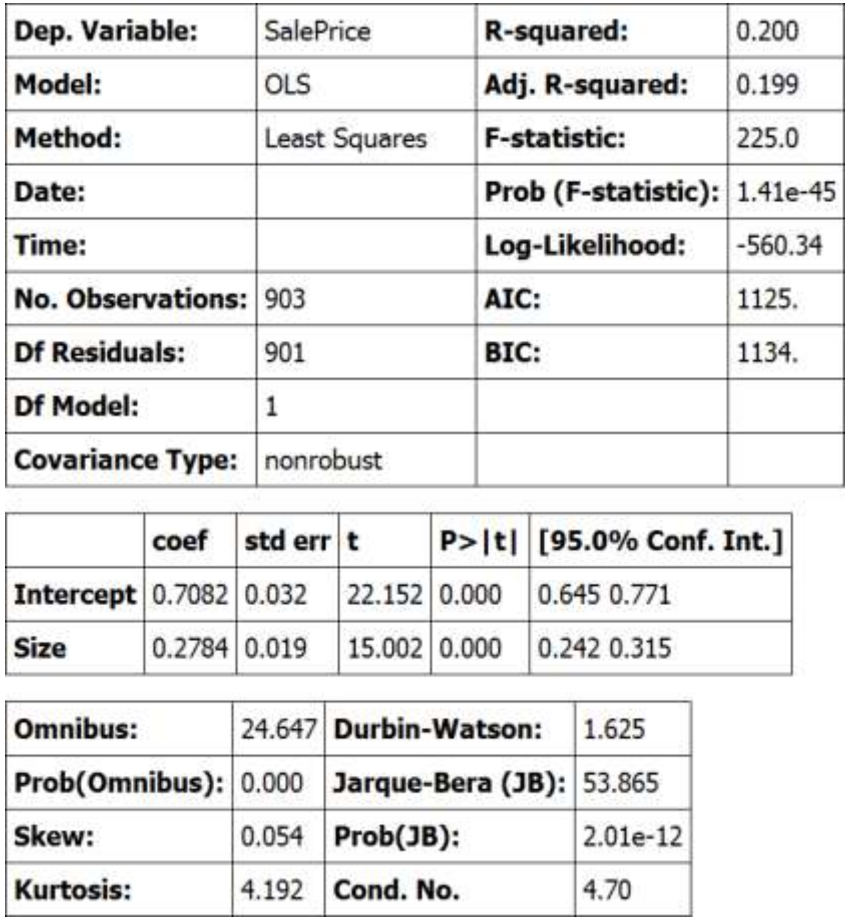
Dep. Variable:	SalePrice	**R-squared:**	0.565
Model:	OLS	**Adj. R-squared:**	0.565
Method:	Least Squares	**F-statistic:**	1255.
Date:		**Prob (F-statistic):**	7.83e-177
Time:		**Log-Likelihood:**	-1689.6
No. Observations:	967	**AIC:**	3381.
Df Residuals:	966	**BIC:**	3386.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Size	0.8176	0.023	35.426	0.000	0.772 0.863

Omnibus:	1830.896	**Durbin-Watson:**	1.722
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	3370566.094
Skew:	13.300	**Prob(JB):**	0.00
Kurtosis:	291.005	**Cond. No.**	1.00

$$SalePrice\ [\$M] = \underbrace{0.}_{\hat{\beta}_0} + \underbrace{.810}_{\hat{\beta}_1} \times Size\ [1{,}000\ sqft]$$

`SalePrice ~ Size` (with outliers removed)



Dep. Variable:	SalePrice	R-squared:	0.200
Model:	OLS	Adj. R-squared:	0.199
Method:	Least Squares	F-statistic:	225.0
Date:		Prob (F-statistic):	1.41e-45
Time:		Log-Likelihood:	-560.34
No. Observations:	903	AIC:	1125.
Df Residuals:	901	BIC:	1134.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	0.7082	0.032	22.152	0.000	0.645 0.771
Size	0.2784	0.019	15.002	0.000	0.242 0.315

Omnibus:	24.647	Durbin-Watson:	1.625
Prob(Omnibus):	0.000	Jarque-Bera (JB):	53.865
Skew:	0.054	Prob(JB):	2.01e-12
Kurtosis:	4.192	Cond. No.	4.70

$$SalePrice\ [\$M] =$$

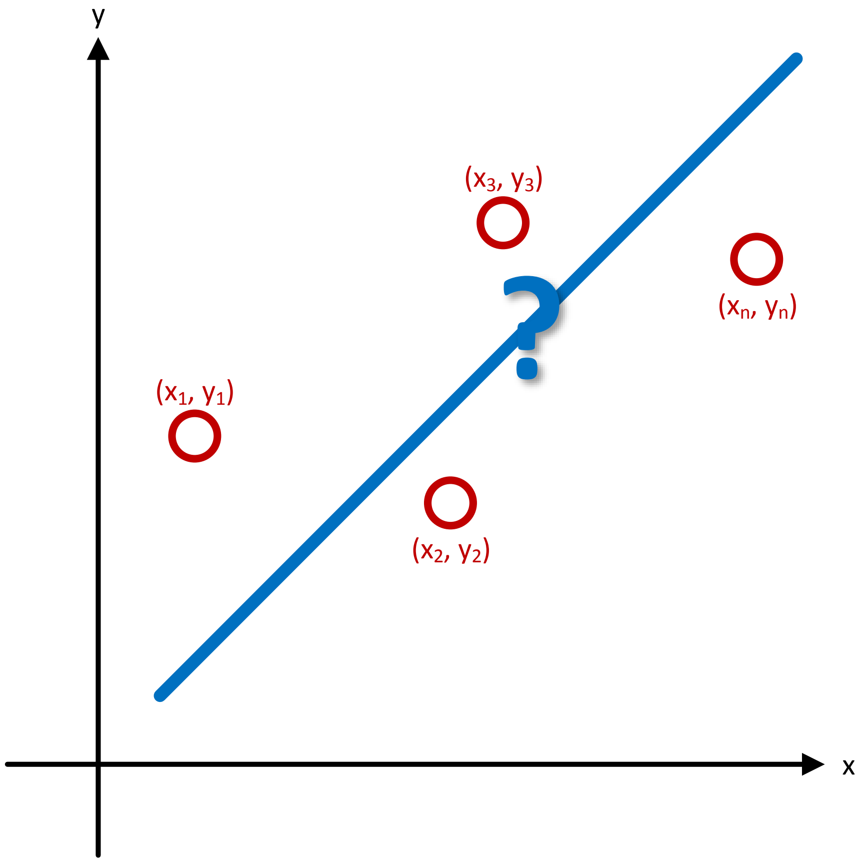
$$\underbrace{.708}_{(was\ .155)} + \underbrace{.278}_{(was\ .750)} \times Size\ [1{,}000\ sqft]$$

(both intercept and slope are now significant)

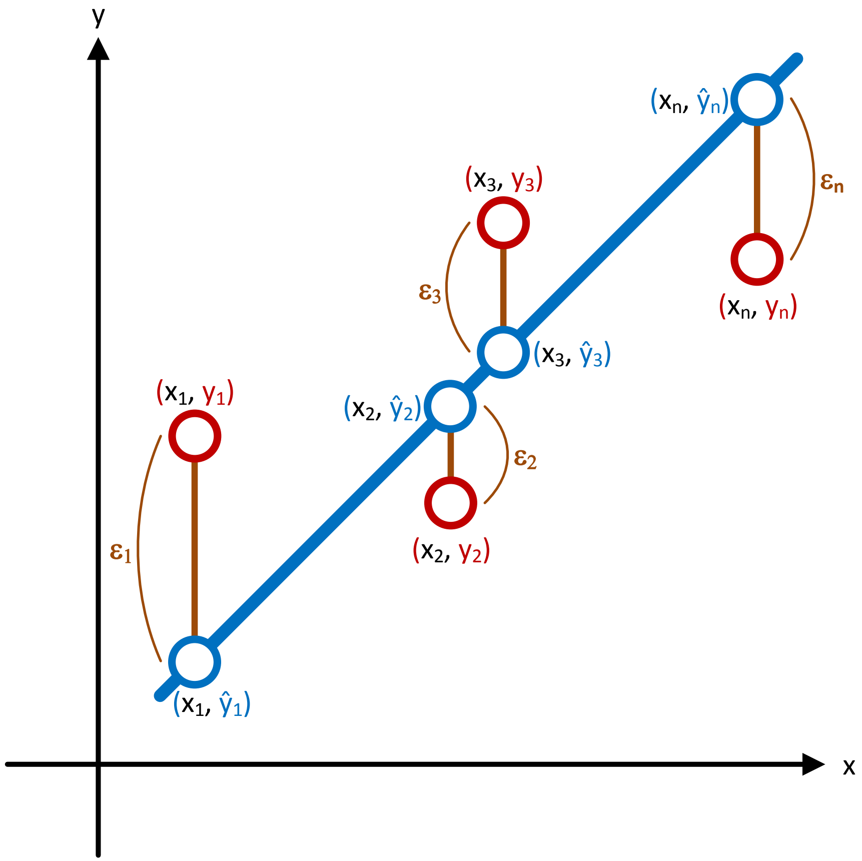
Simple Linear Regression

How is a regression model fitted to a dataset?

How do we estimate $\hat{\beta}_0$ and $\hat{\beta}_1$?



We can estimate $\hat{\beta}_0$ and $\hat{\beta}_1$ with Ordinary Least Squares (OLS)



- Hypothesis

$$y_{\beta_0,\beta_1}(x) = \beta_0 + \beta_1 \cdot x$$

- Parameters

$$\beta_0, \beta_1$$

- Loss Function

$$L\big(y_i, y(x_i)\big) = \big(y_i - y(x_i)\big)^2 = \varepsilon_i^2$$

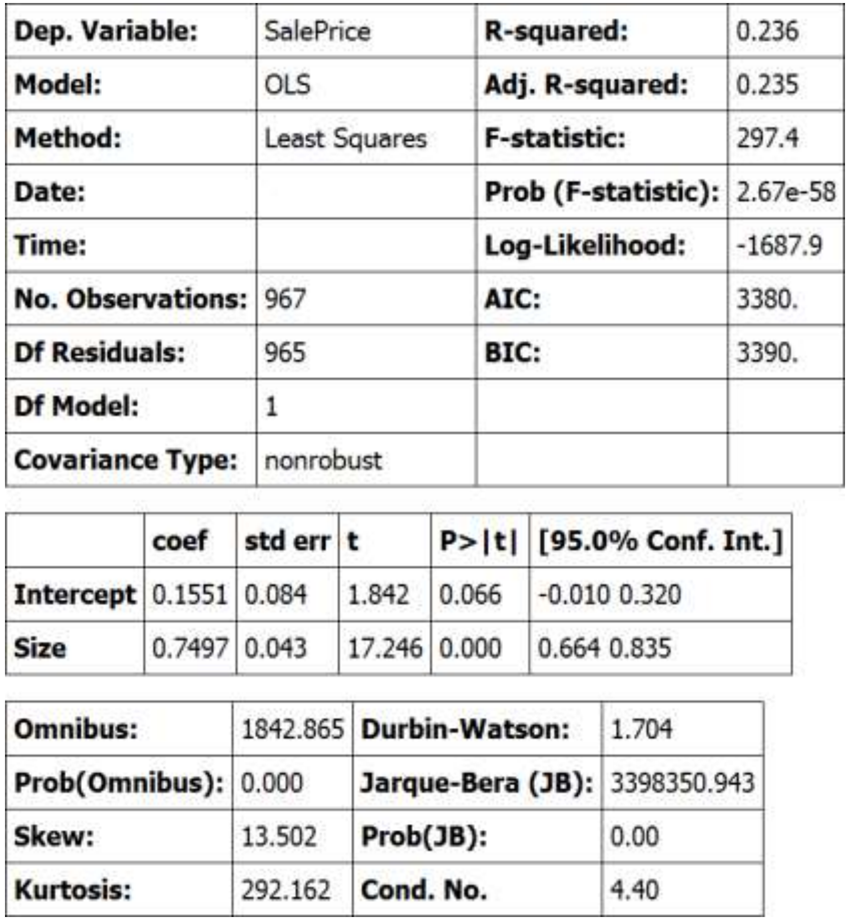
$$L(\beta_0, \beta_1) = \sum_{i=1}^{m} L\big(y_i, y(x_i)\big) = \|\varepsilon\|^2$$

- Goal

$$\min_{\beta_0,\beta_1} L(\beta_0, \beta_1) \text{ or } \min_{\beta_0,\beta_1} \|\varepsilon\|^2$$

(i.e., minimizing the least squares)

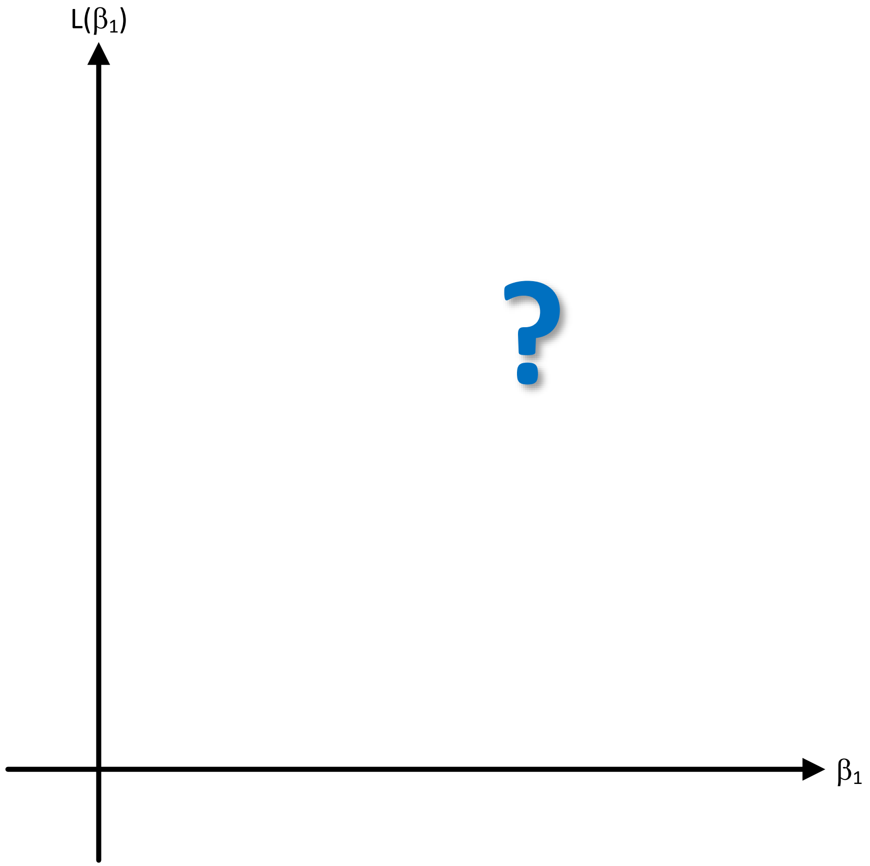
What's the shape of the loss function L and what's the optimal $\hat{\beta}_0$ and $\hat{\beta}_1$? For the time being, let's set $\hat{\beta}_0$ to .1551, draw L as a function of $\hat{\beta}_1$ only, and find the optimal $\hat{\beta}_1$



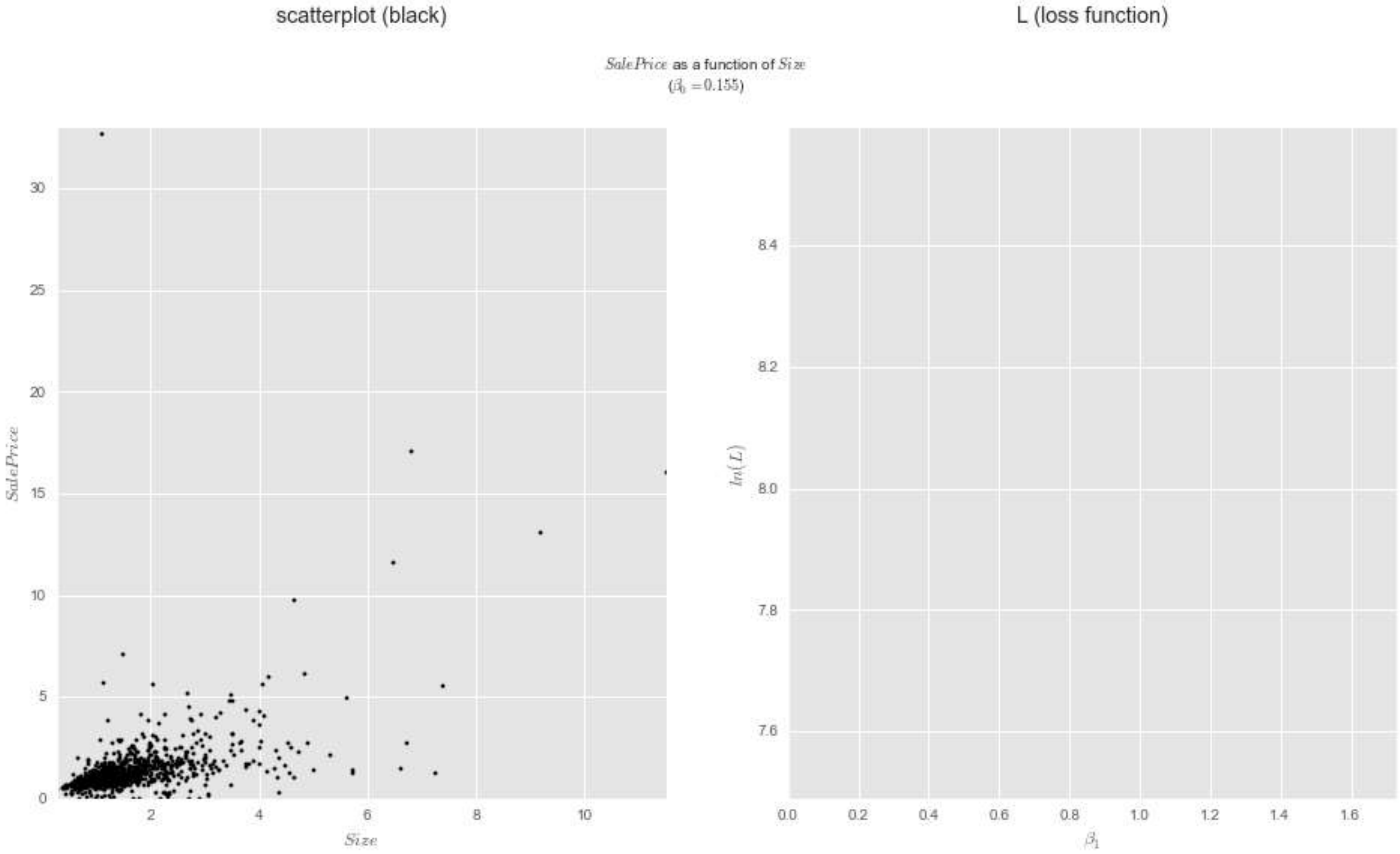
Dep. Variable:	SalePrice	**R-squared:**	0.236
Model:	OLS	**Adj. R-squared:**	0.235
Method:	Least Squares	**F-statistic:**	297.4
Date:		**Prob (F-statistic):**	2.67e-58
Time:		**Log-Likelihood:**	-1687.9
No. Observations:	967	**AIC:**	3380.
Df Residuals:	965	**BIC:**	3390.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	0.1551	0.084	1.842	0.066	-0.010 0.320
Size	0.7497	0.043	17.246	0.000	0.664 0.835

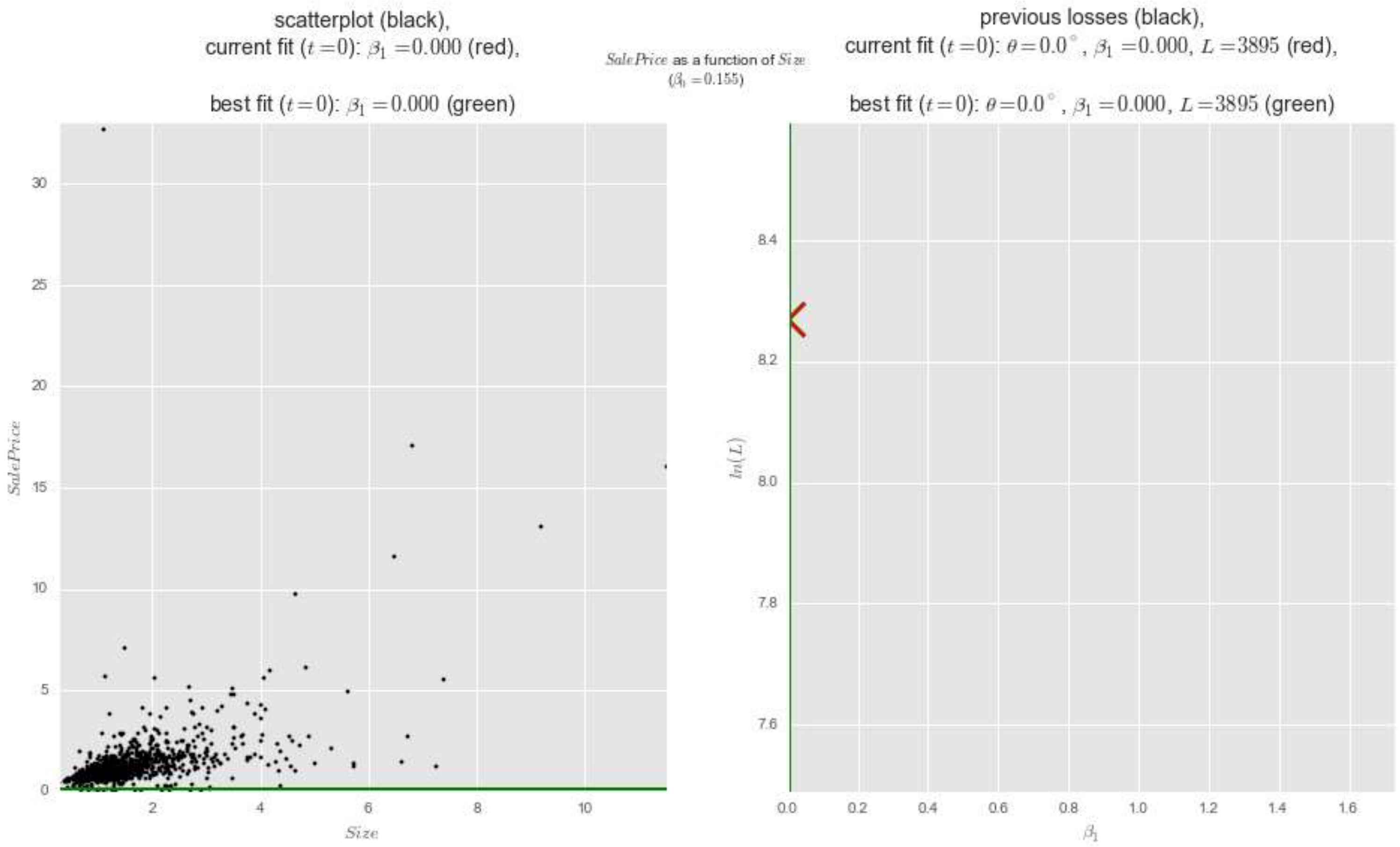
Omnibus:	1842.865	**Durbin-Watson:**	1.704
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	3398350.943
Skew:	13.502	**Prob(JB):**	0.00
Kurtosis:	292.162	**Cond. No.**	4.40



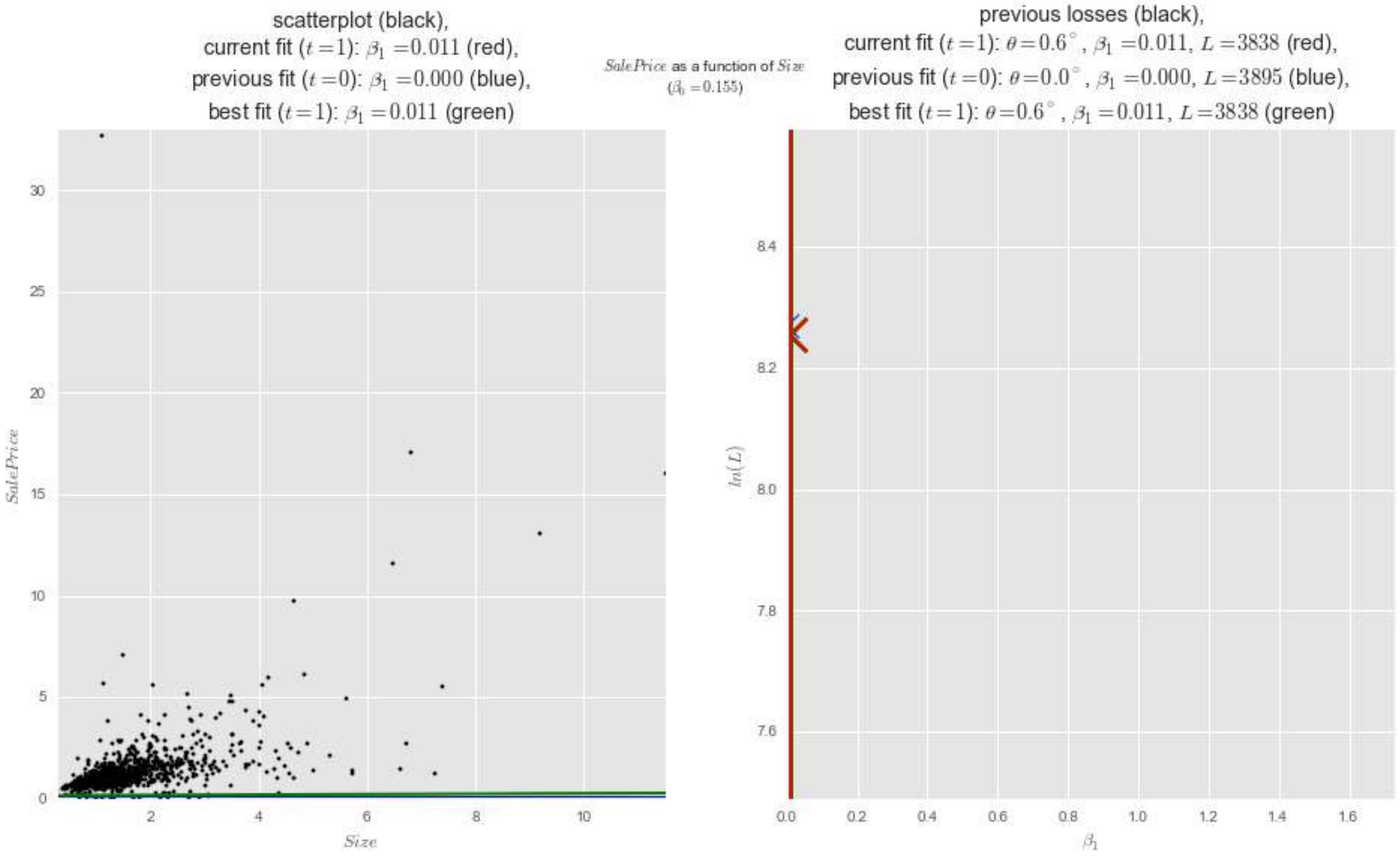
Demo | On the left, the scatterplot of our data and the fitted lines at different angles θ ($\beta_1 = \tan(\theta)$). On the right $L(\beta_1)$



Demo | $t = 0$


scatterplot (black),
current fit ($t = 0$): $\beta_1 = 0.000$ (red),
best fit ($t = 0$): $\beta_1 = 0.000$ (green)
$SalePrice$ as a function of $Size$
($\beta_0 = 0.155$)
previous losses (black),
current fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (red),
best fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (green)
$SalePrice$
30
25
20
15
10
5
0
2
4
6
8
10
$Size$
$ln(L)$
8.4
8.2
8.0
7.8
7.6
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
β_1

Demo | $t = 1$

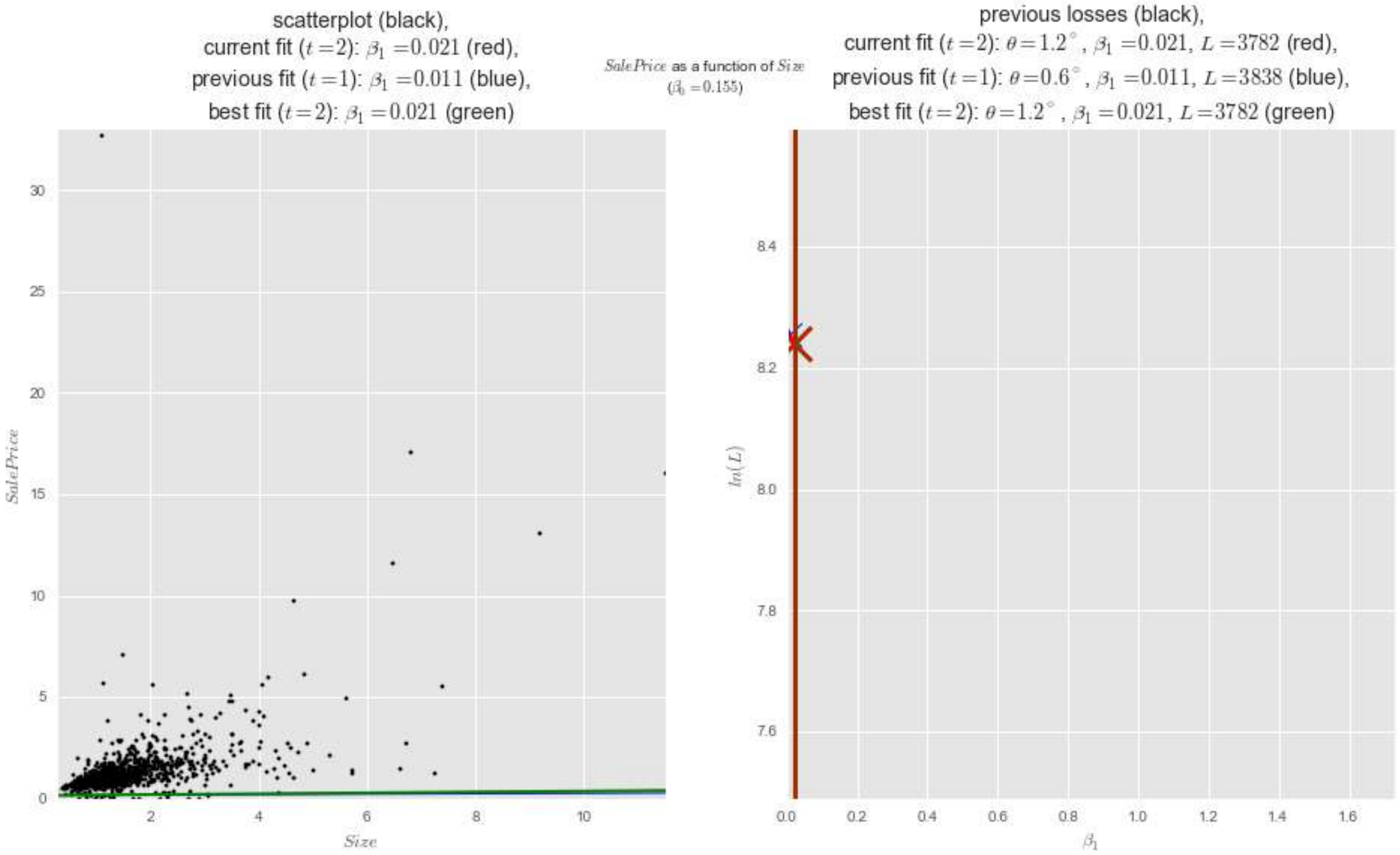


$SalePrice$ as a function of $Size$
($\beta_0 = 0.155$)

scatterplot (black),
current fit ($t = 1$): $\beta_1 = 0.011$ (red),
previous fit ($t = 0$): $\beta_1 = 0.000$ (blue),
best fit ($t = 1$): $\beta_1 = 0.011$ (green)

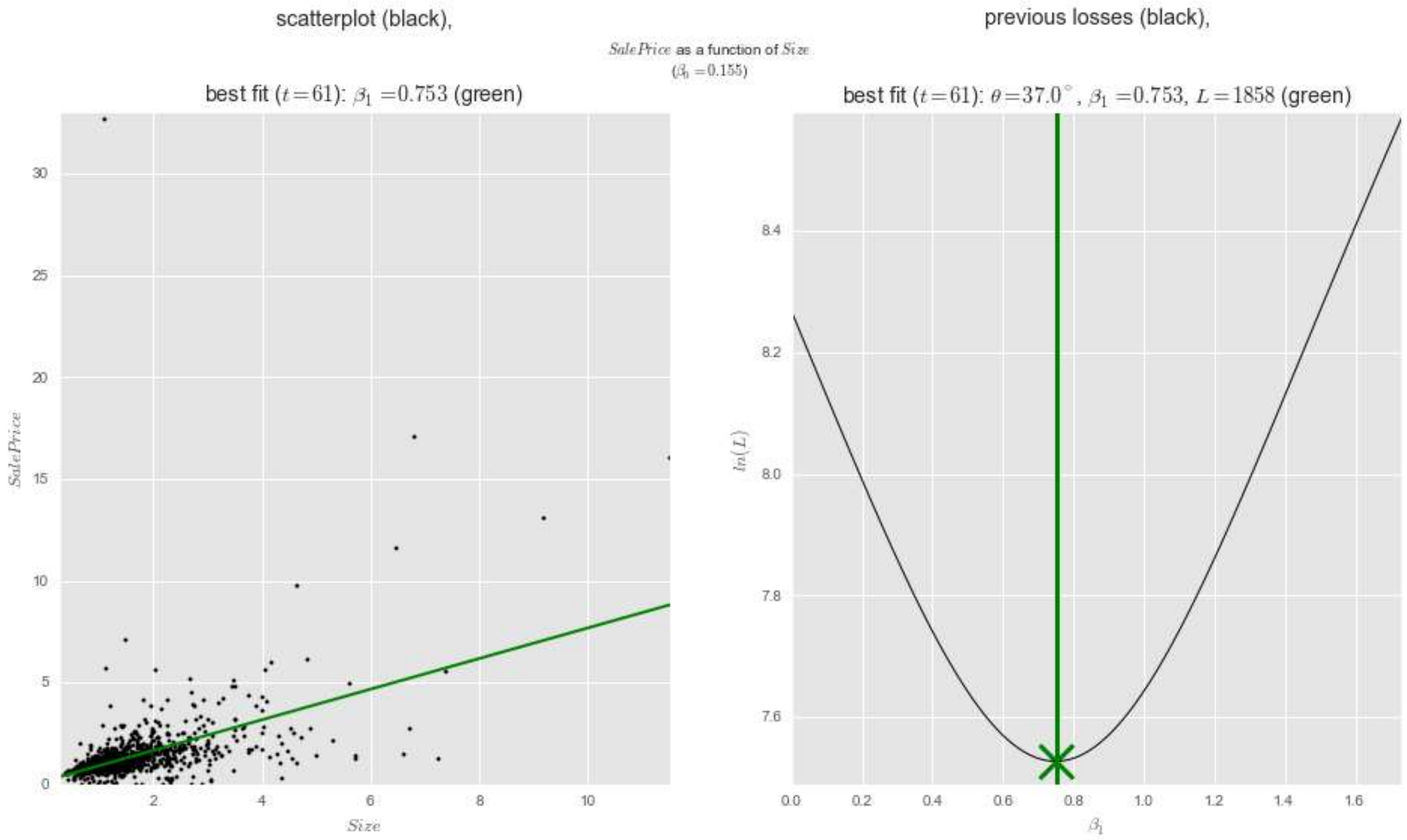
previous losses (black),
current fit ($t = 1$): $\theta = 0.6^\circ$, $\beta_1 = 0.011$, $L = 3838$ (red),
previous fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (blue),
best fit ($t = 1$): $\theta = 0.6^\circ$, $\beta_1 = 0.011$, $L = 3838$ (green)

Demo | $t = 2$

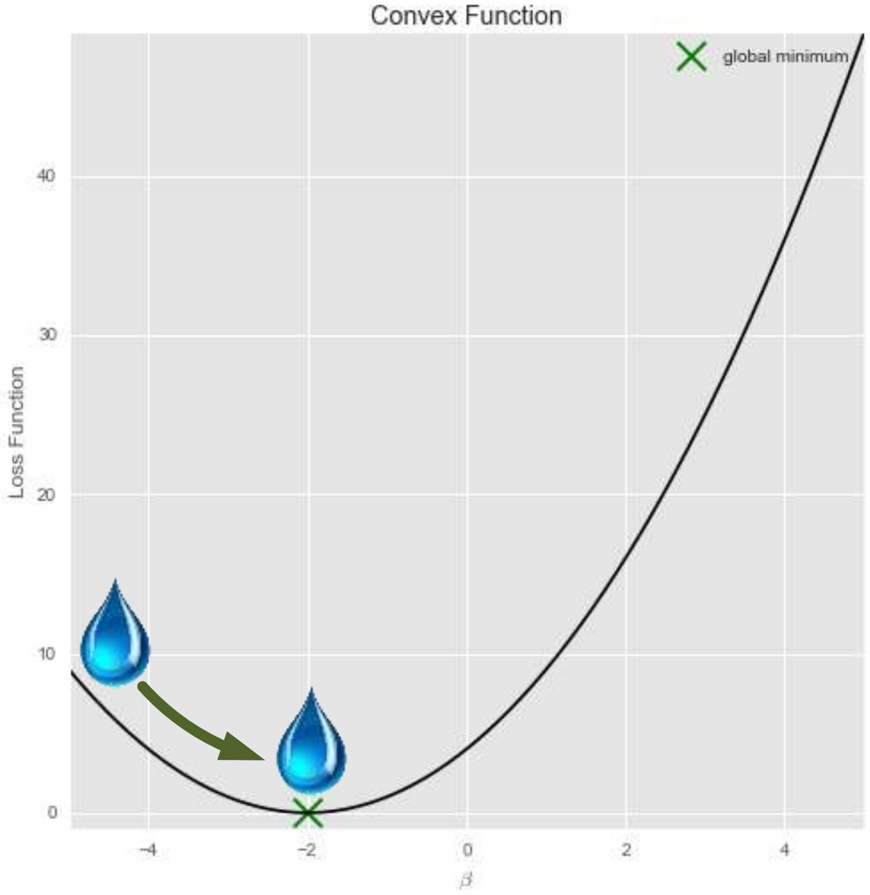


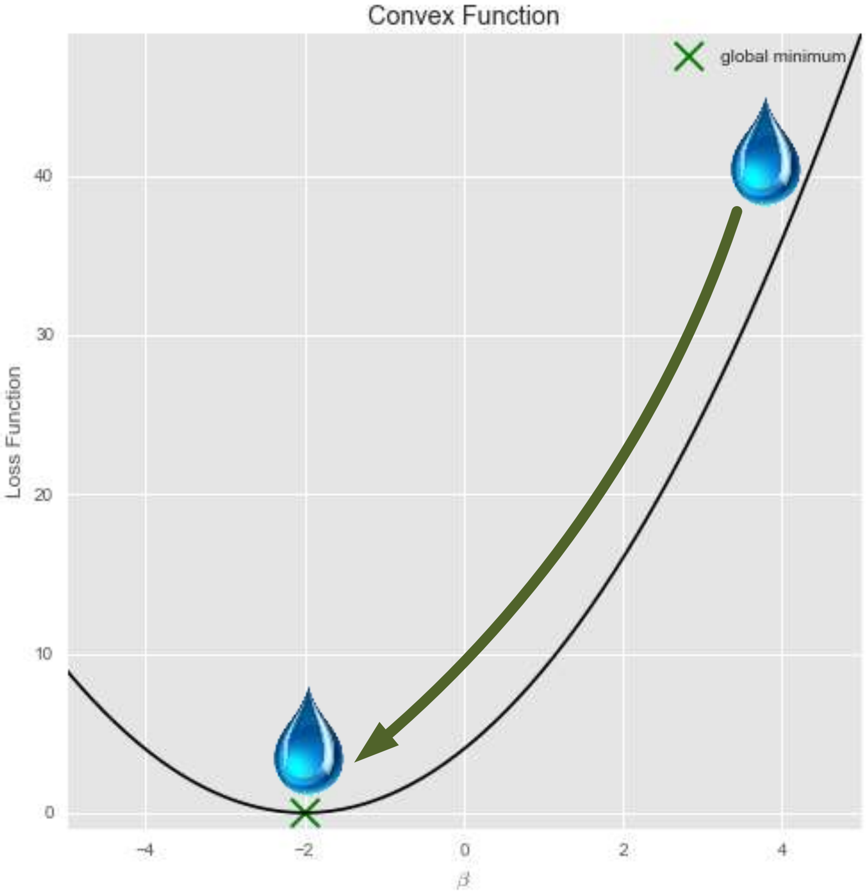
$SalePrice$ as a function of $Size$
($\beta_0 = 0.155$)
scatterplot (black),
current fit ($t = 2$): $\beta_1 = 0.021$ (red),
previous fit ($t = 1$): $\beta_1 = 0.011$ (blue),
best fit ($t = 2$): $\beta_1 = 0.021$ (green)
$SalePrice$
30
25
20
15
10
5
0
2
4
6
8
10
$Size$
previous losses (black),
current fit ($t = 2$): $\theta = 1.2^\circ$, $\beta_1 = 0.021$, $L = 3782$ (red),
previous fit ($t = 1$): $\theta = 0.6^\circ$, $\beta_1 = 0.011$, $L = 3838$ (blue),
best fit ($t = 2$): $\theta = 1.2^\circ$, $\beta_1 = 0.021$, $L = 3782$ (green)
$ln(L)$
8.4
8.2
8.0
7.8
7.6
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
β_1

Demo | On the left, the best fitted line with $\hat{\beta}_1 = .753$. On the right $L(\beta_1)$

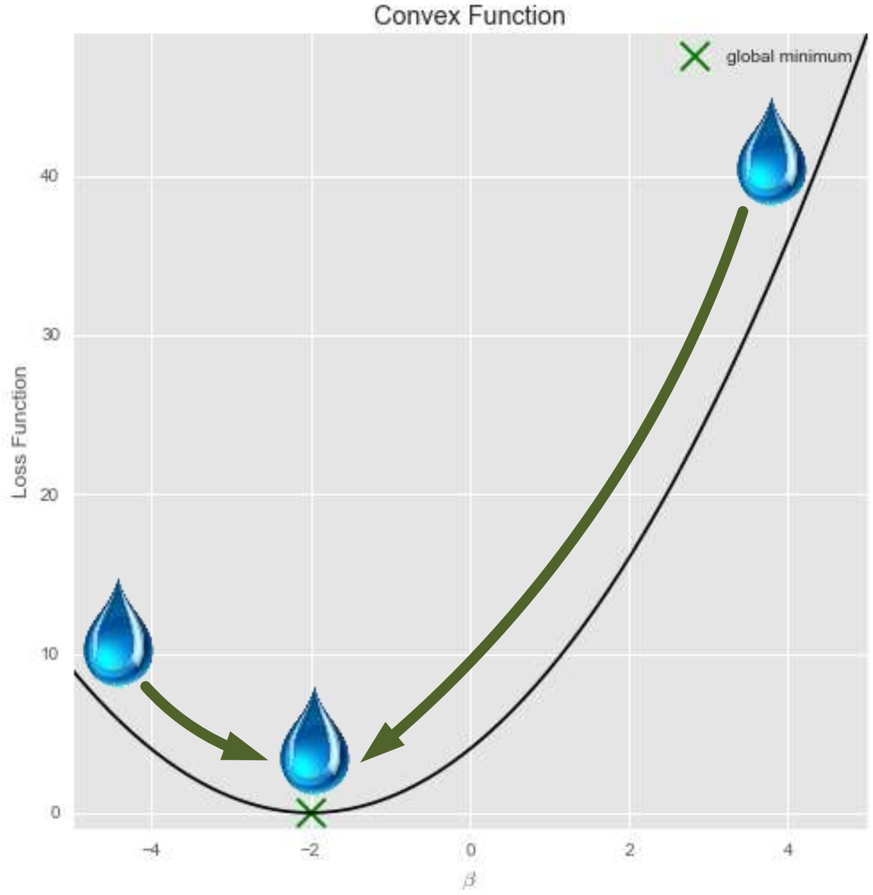


Look at the following curve, where would these two drop of water go? (cont.)

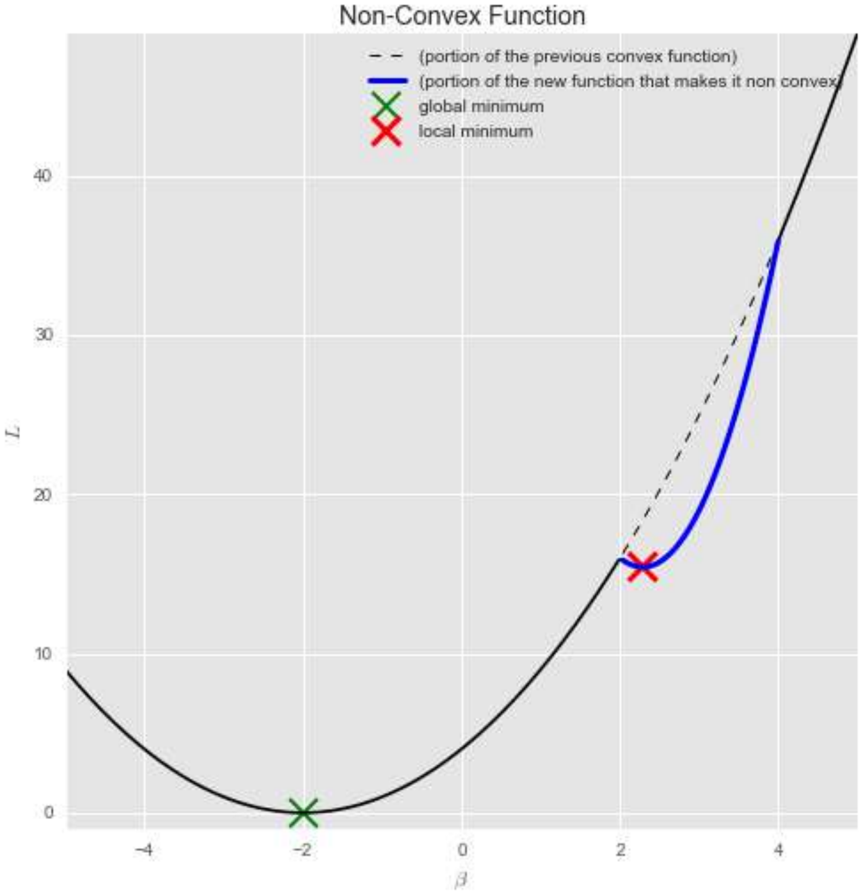




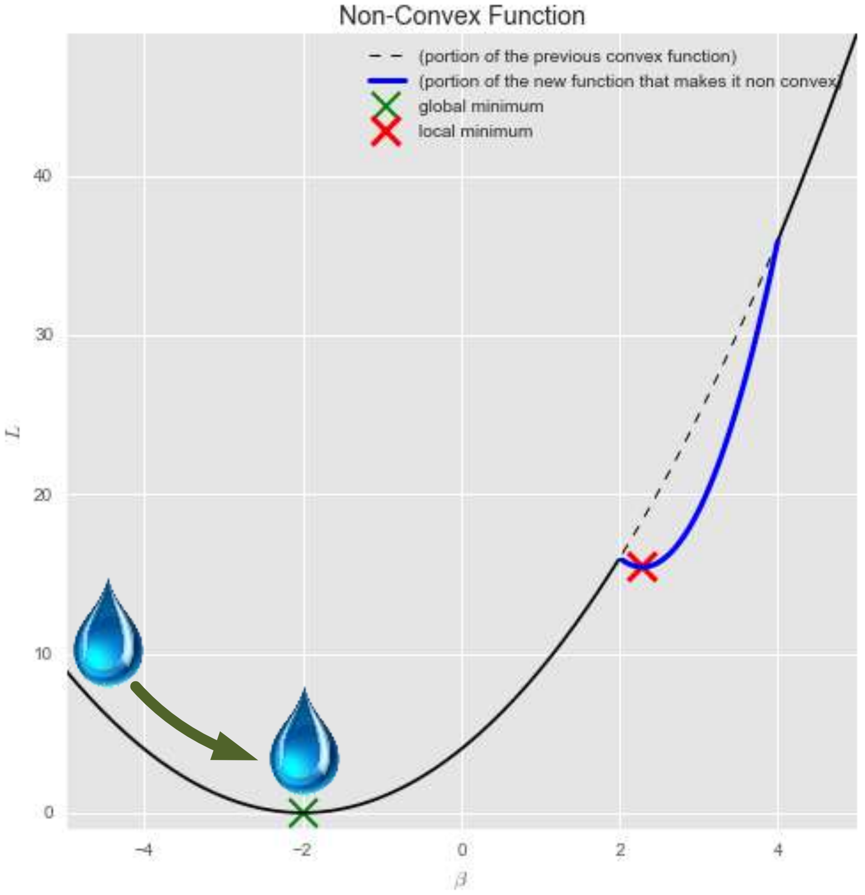
$L(\beta_1)$ is a convex function: It reaches a minimum value only once (global minimum) and following the gravity (greatest slope), a drop of water placed anywhere along the curve would descent towards that minimum

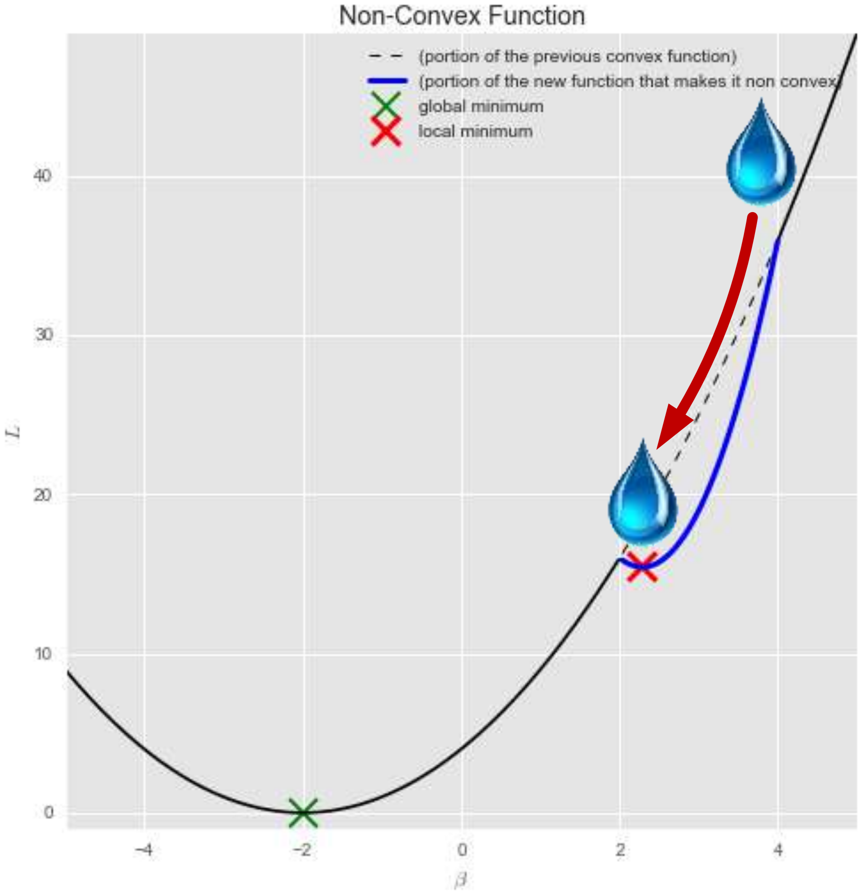


In contrast, here’s an example of non-convex function: The function has a global minimum but also another (local) minimum



The drops of water can go down to the global minimum or get stuck in a local minimum





Reaching the global minimum following the greatest slope is the idea behind gradient descent (assuming a convex function)

- Goal

$$\min_{\beta_1} L(\beta_1)$$

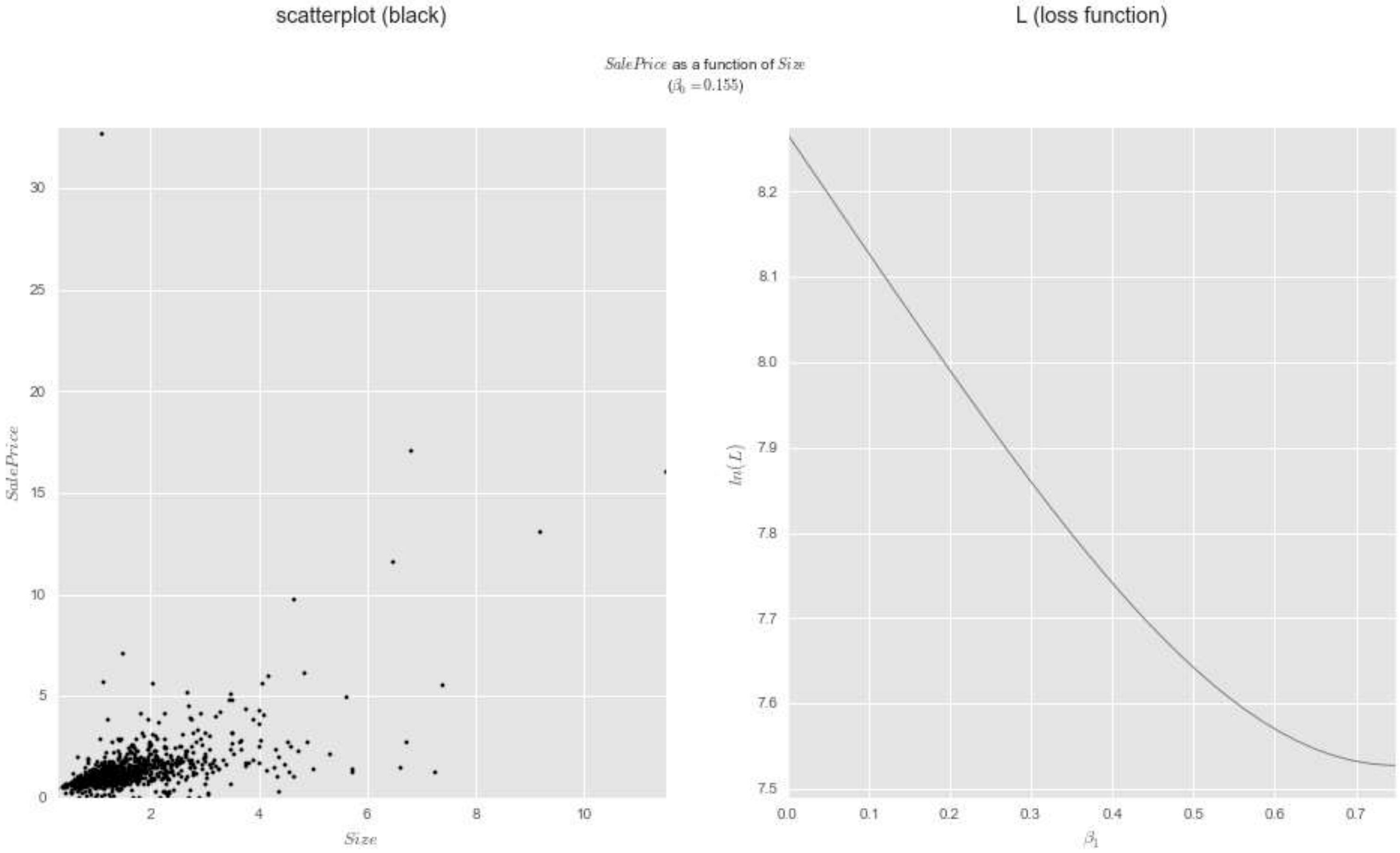
(i.e., minimizing the least squares)

- Gradient Descent Algorithm
  - Start with some β_1, e.g., $\beta_1 = 0$
  - Repeat until convergence

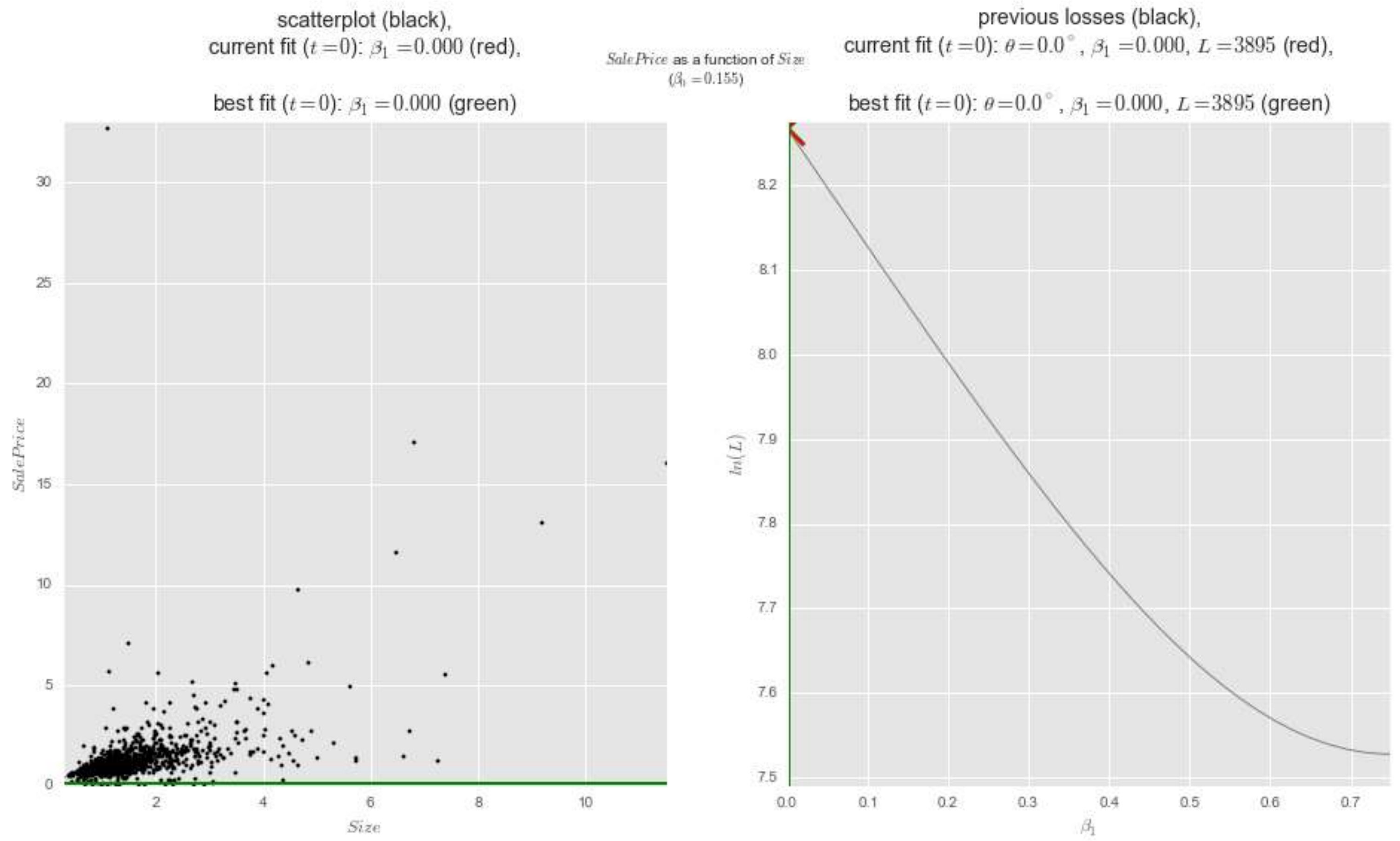
$$\beta_1 \coloneqq \beta_1 - \alpha \underbrace{\frac{\partial}{\partial \beta_1} L(\beta_1)}_{\frac{1}{m}\sum_{i=1}^{m} x_i \cdot \underbrace{(\beta_0 + \beta_1 \cdot x_i - y_i)}_{-\varepsilon_i}} = \beta_1 + \frac{\alpha}{m} \sum_{i=1}^{m} \varepsilon_i \cdot x_i$$

```
def y_hat(beta_0_hat, beta_1_hat, x):
    return beta_0_hat + beta_1_hat * x

def L(y_hat):
    return sum((y - y_hat) ** 2)

beta_1 = 0

for _ in range(n):
    beta_1 += alpha *\
        ((y - y_hat(beta_0, beta_1, x)) * x).\
            mean()
```

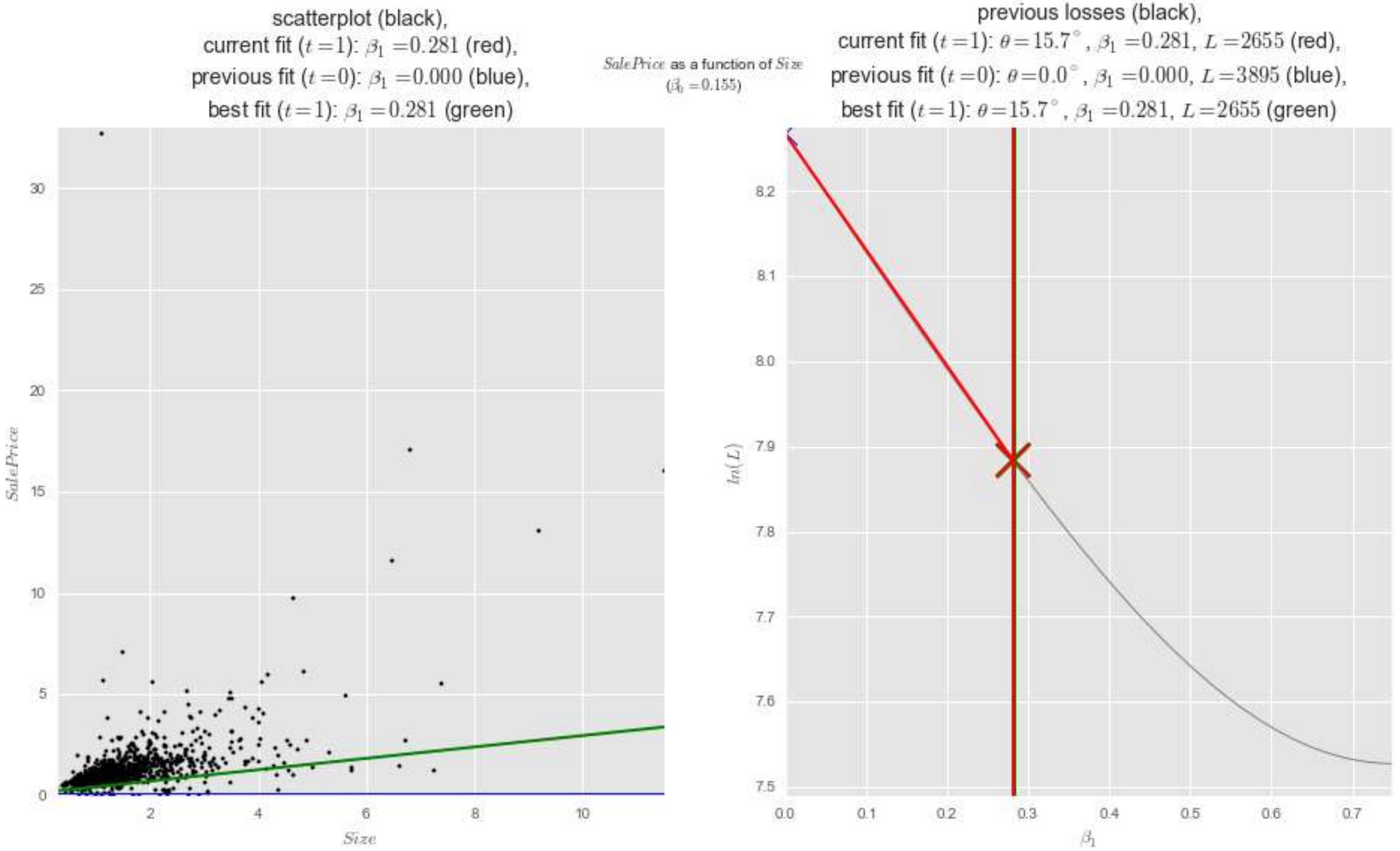
Demo | On the left, the scatterplot of our data and the fitted lines at different angles θ ($\beta_1 = \tan(\theta)$). On the right $L(\beta_1)$. Note: we show the shape of the loss function but the gradient descent algorithm doesn't know it…



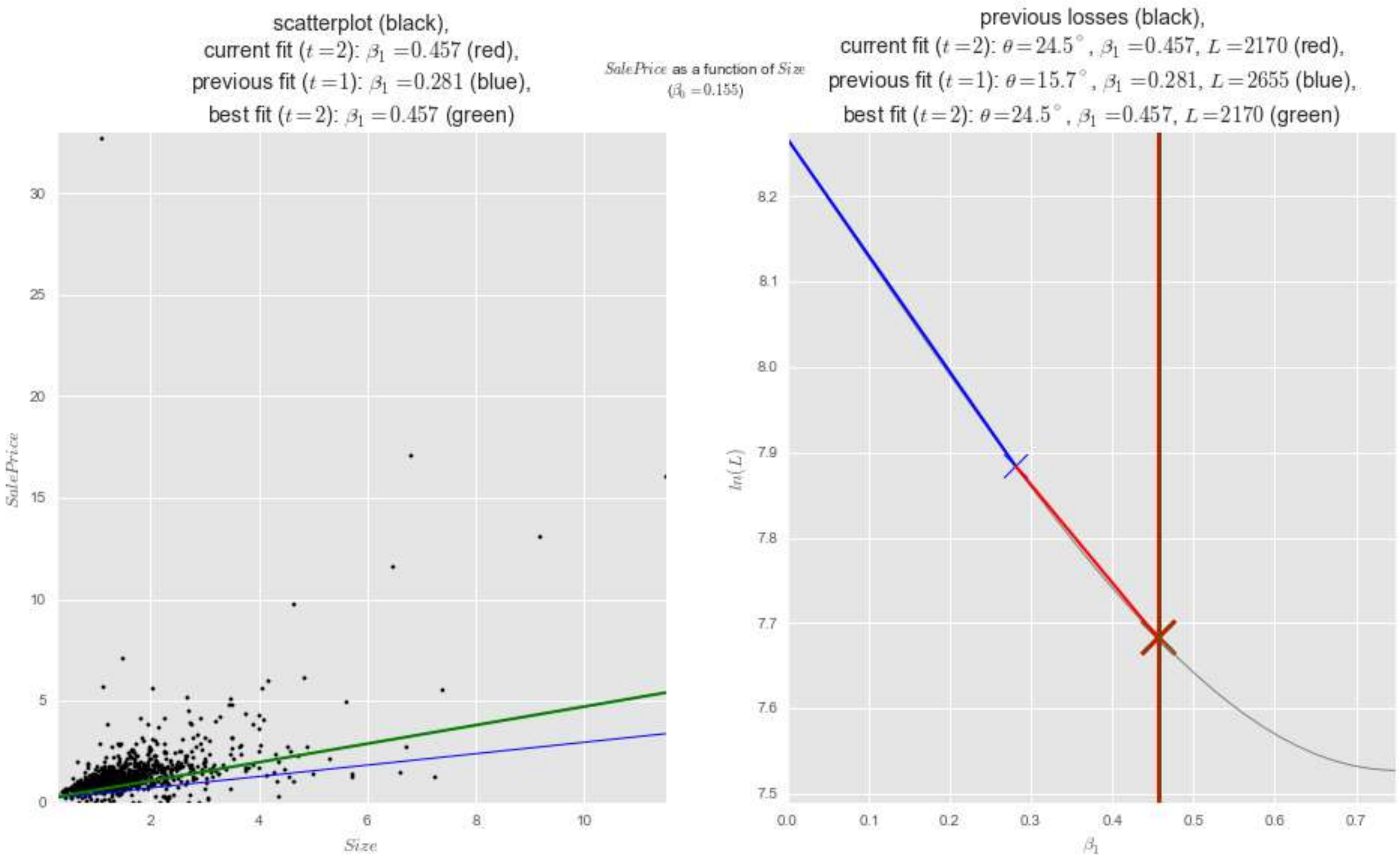
Demo | $t = 0$


scatterplot (black),
current fit ($t = 0$): $\beta_1 = 0.000$ (red),
best fit ($t = 0$): $\beta_1 = 0.000$ (green)
$SalePrice$ as a function of $Size$
($\beta_1 = 0.155$)
previous losses (black),
current fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (red),
best fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (green)
$SalePrice$
30
25
20
15
10
5
0
2
4
6
8
10
$Size$
$ln(L)$
8.2
8.1
8.0
7.9
7.8
7.7
7.6
7.5
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
β_1

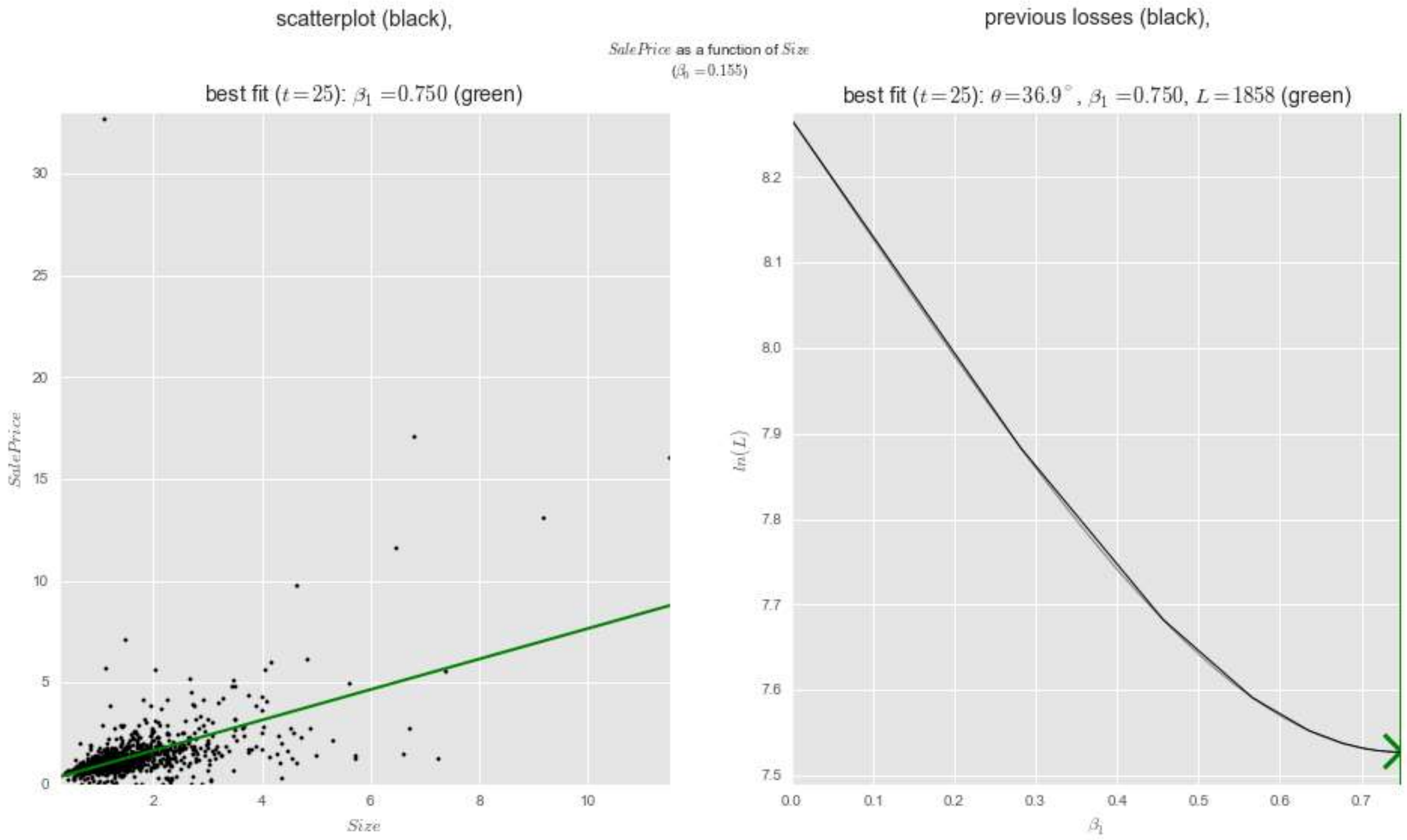
Demo | $t = 1$ | The gradient descent made a "big jump"



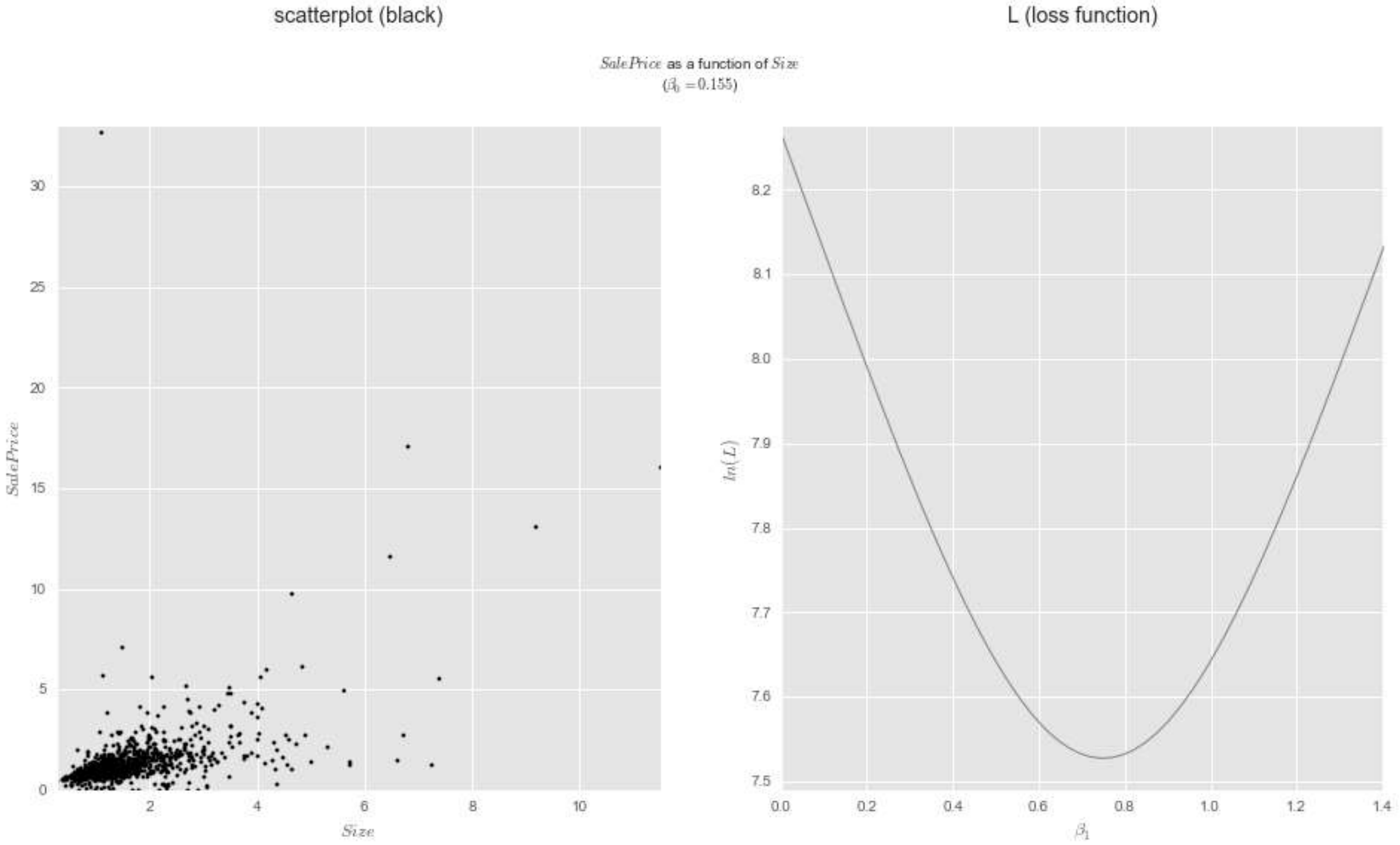
Demo | $t = 2$ | Another jump although not as big; the estimated $\hat{\beta}_1$ is getting close to the optimal pretty fast



Demo | On the left, the best fitted line with $\hat{\beta}_1 = .750$. On the right the points of $L(\beta_1)$ following the gradient descent



Demo | α is the learning rate. In the previous example it was set to .25. The idea is to only incorporate a fraction of the learning. But what happens if α is too high? Let's try $\alpha = .5$…

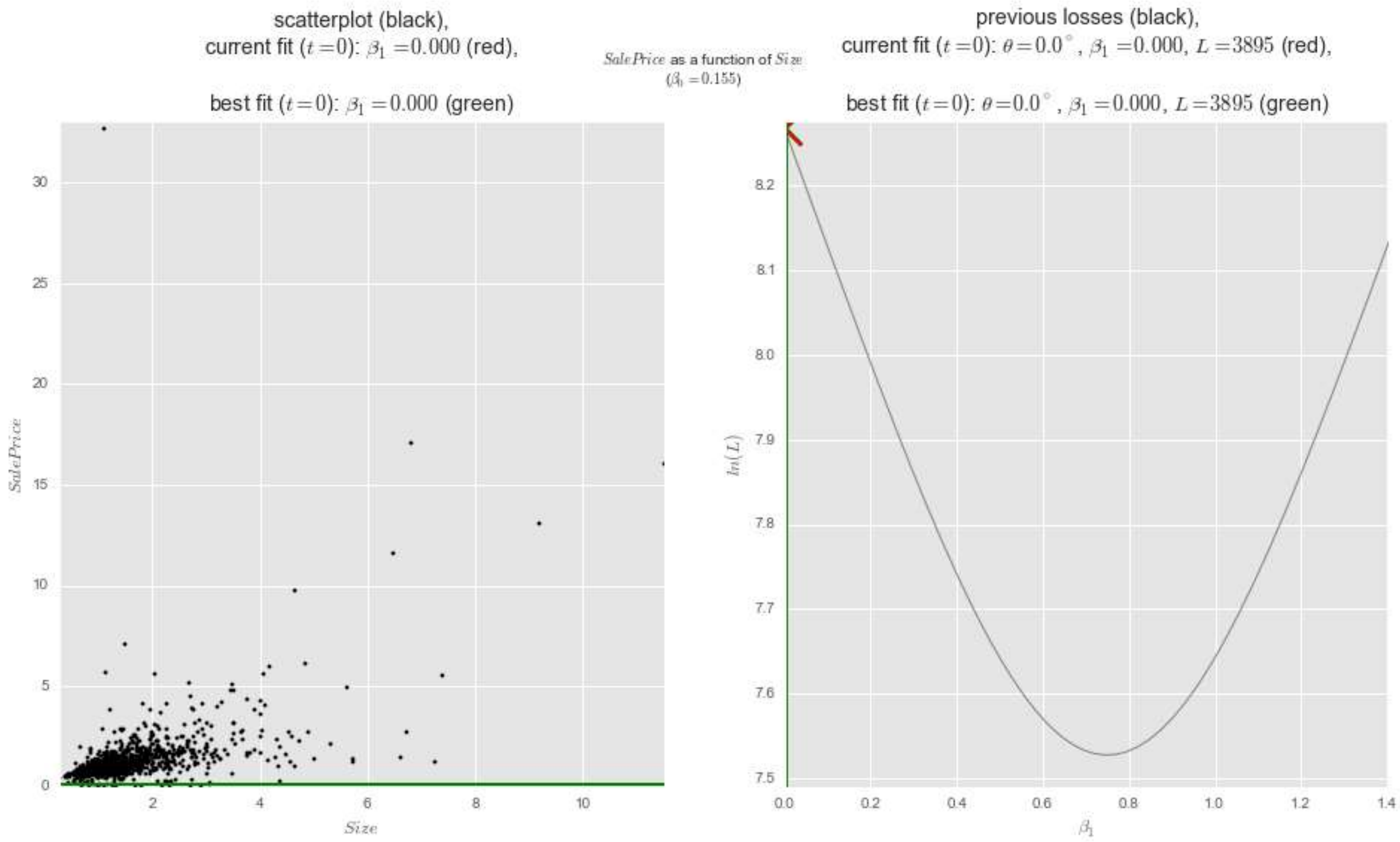


scatterplot (black)

L (loss function)

$SalePrice$ as a function of $Size$
$(\beta_0 = 0.155)$

Demo | $t = 0$



$SalePrice$ as a function of $Size$ ($\beta_1 = 0.155$)

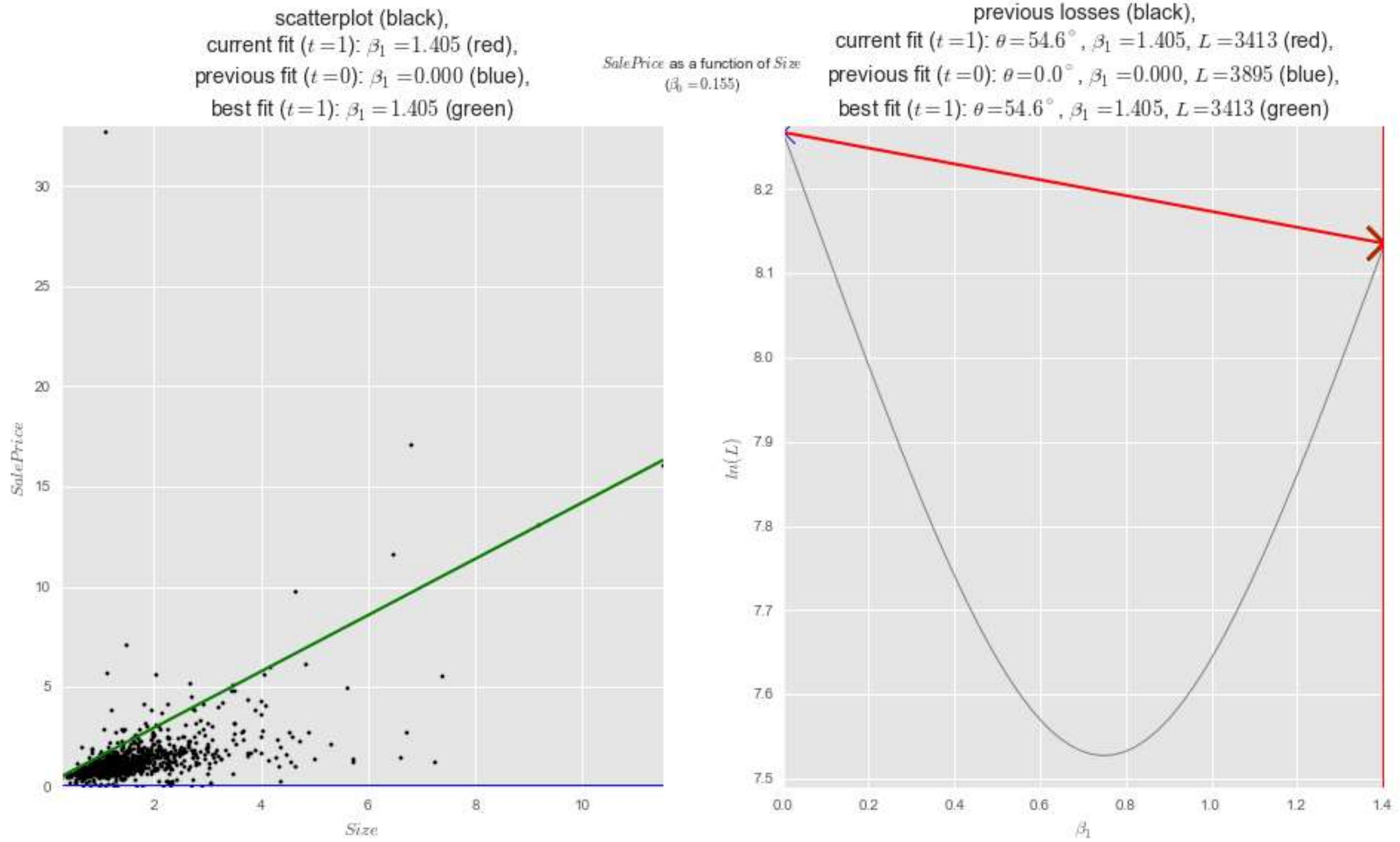
scatterplot (black),
current fit ($t = 0$): $\beta_1 = 0.000$ (red),

best fit ($t = 0$): $\beta_1 = 0.000$ (green)

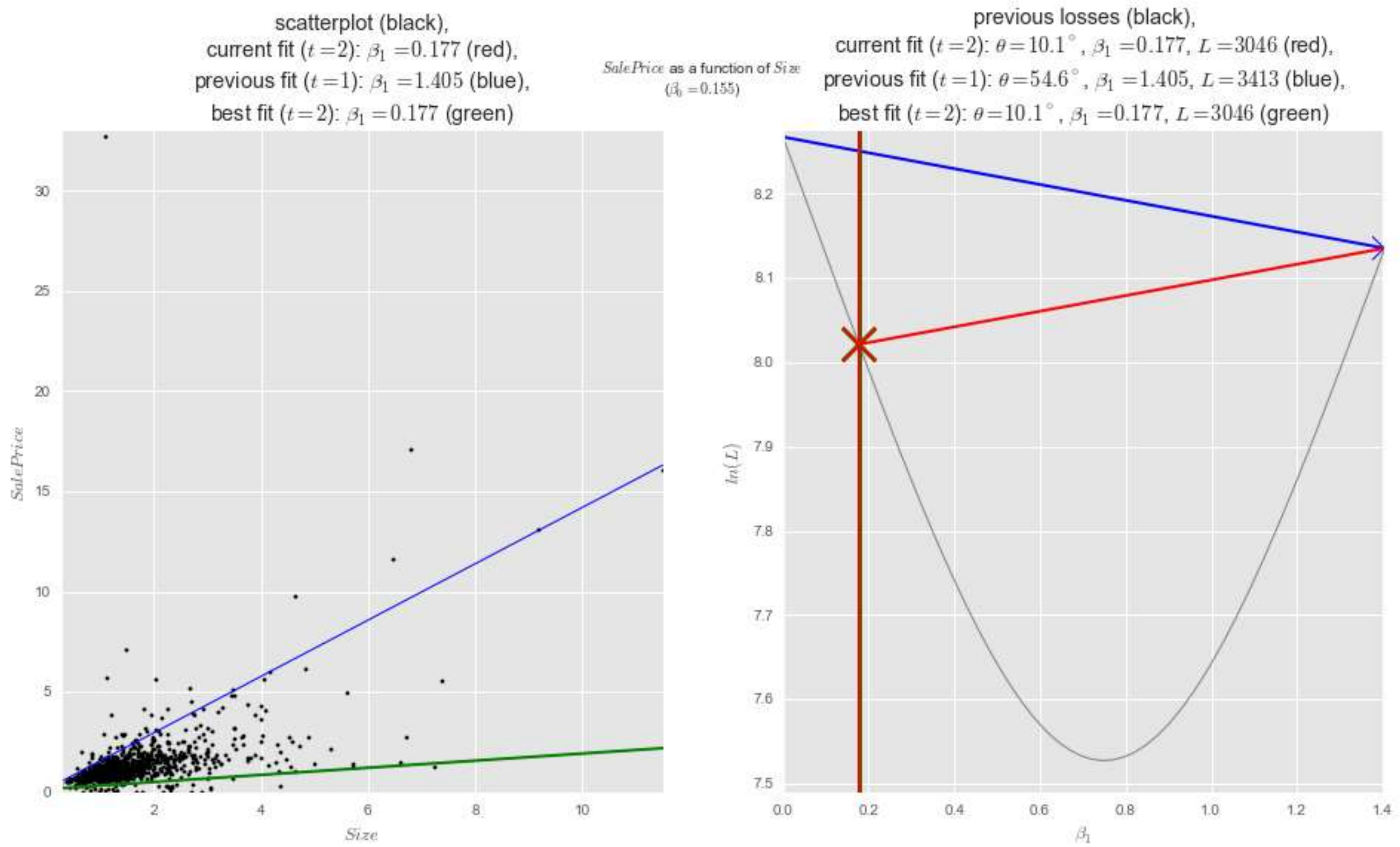
previous losses (black),
current fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (red),

best fit ($t = 0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (green)

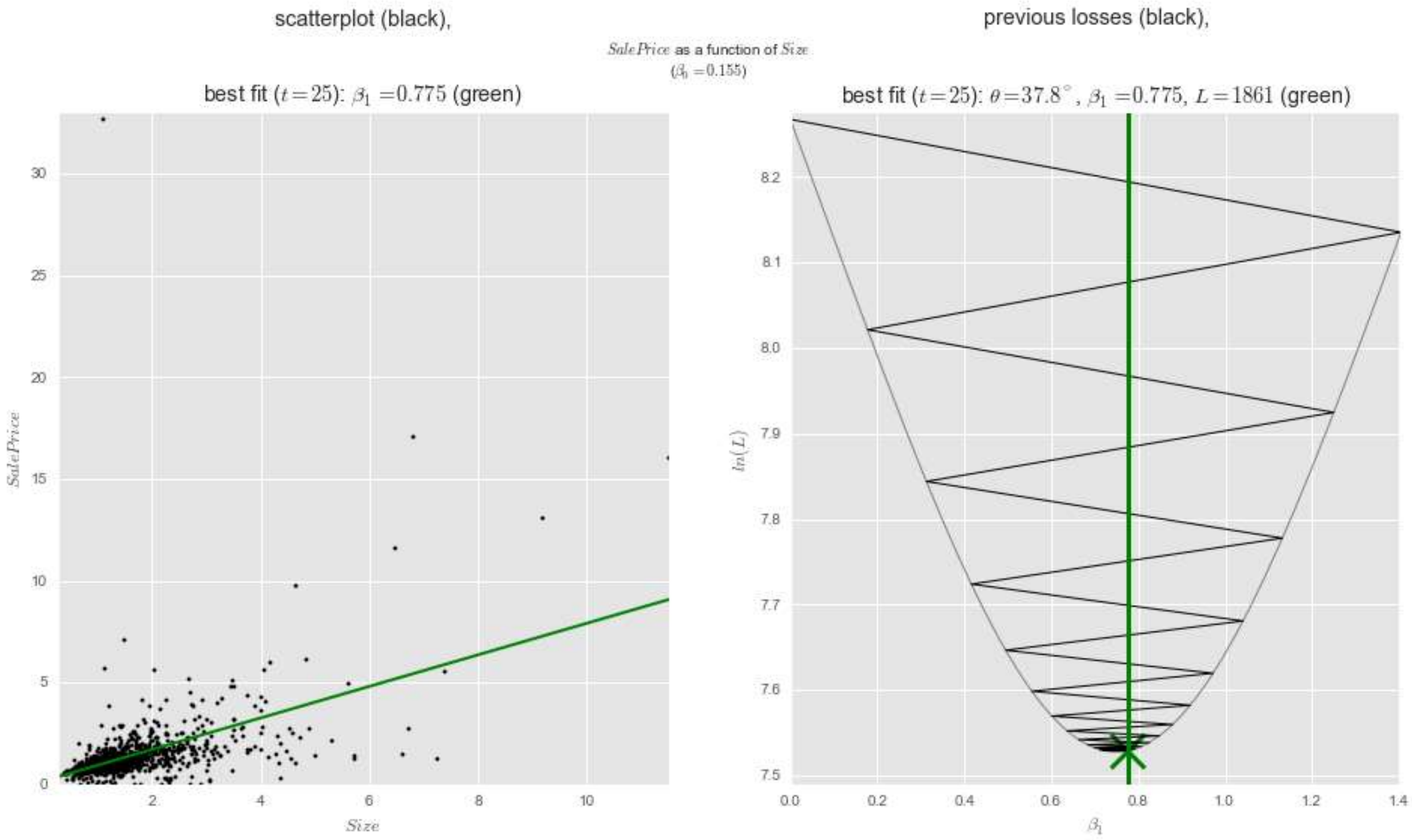
Demo | $t = 1$ | The gradient descent “overshoot”



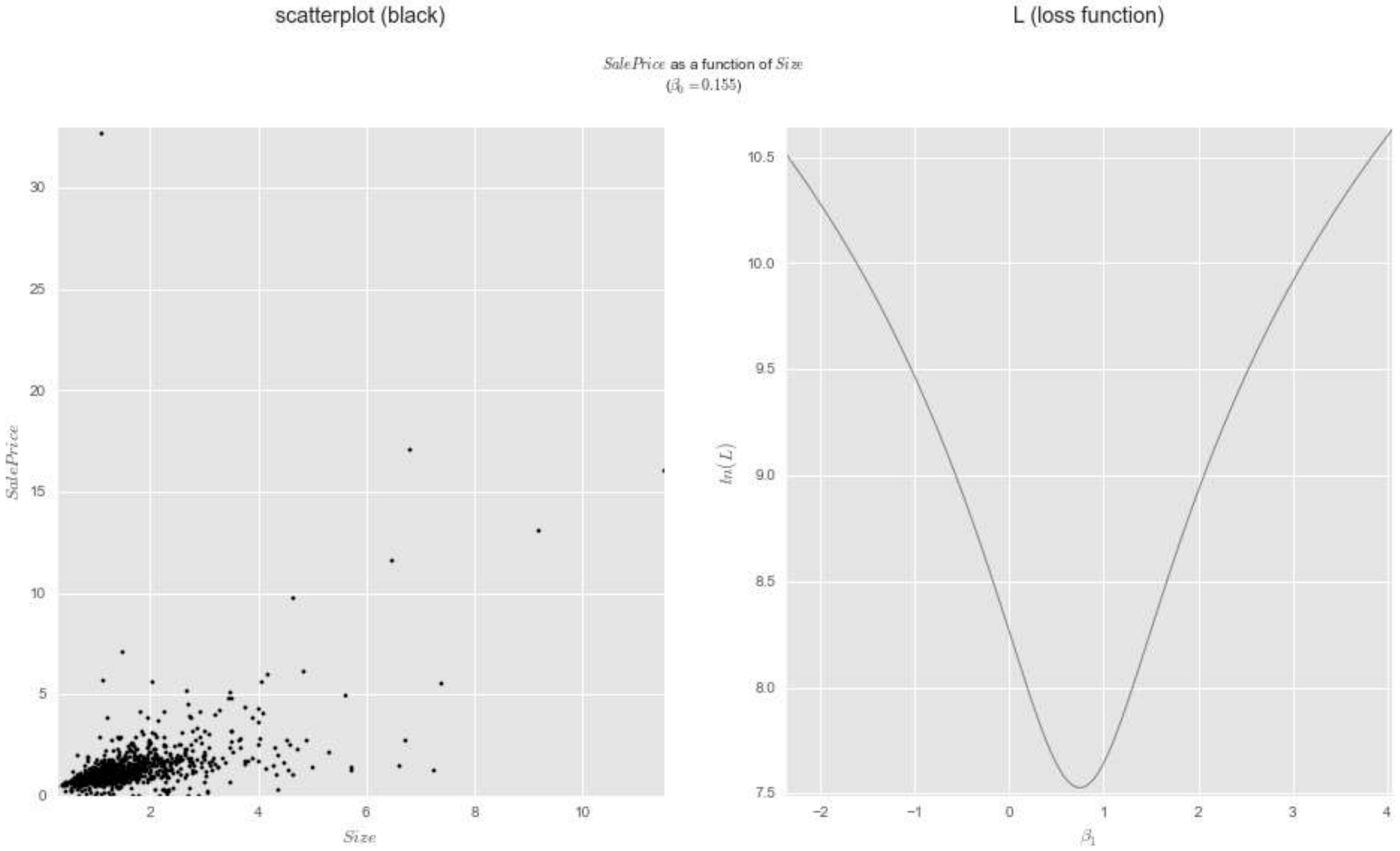
Demo | $t = 2$ | We overshoot again but it seems that we are converging (albeit slowly)…



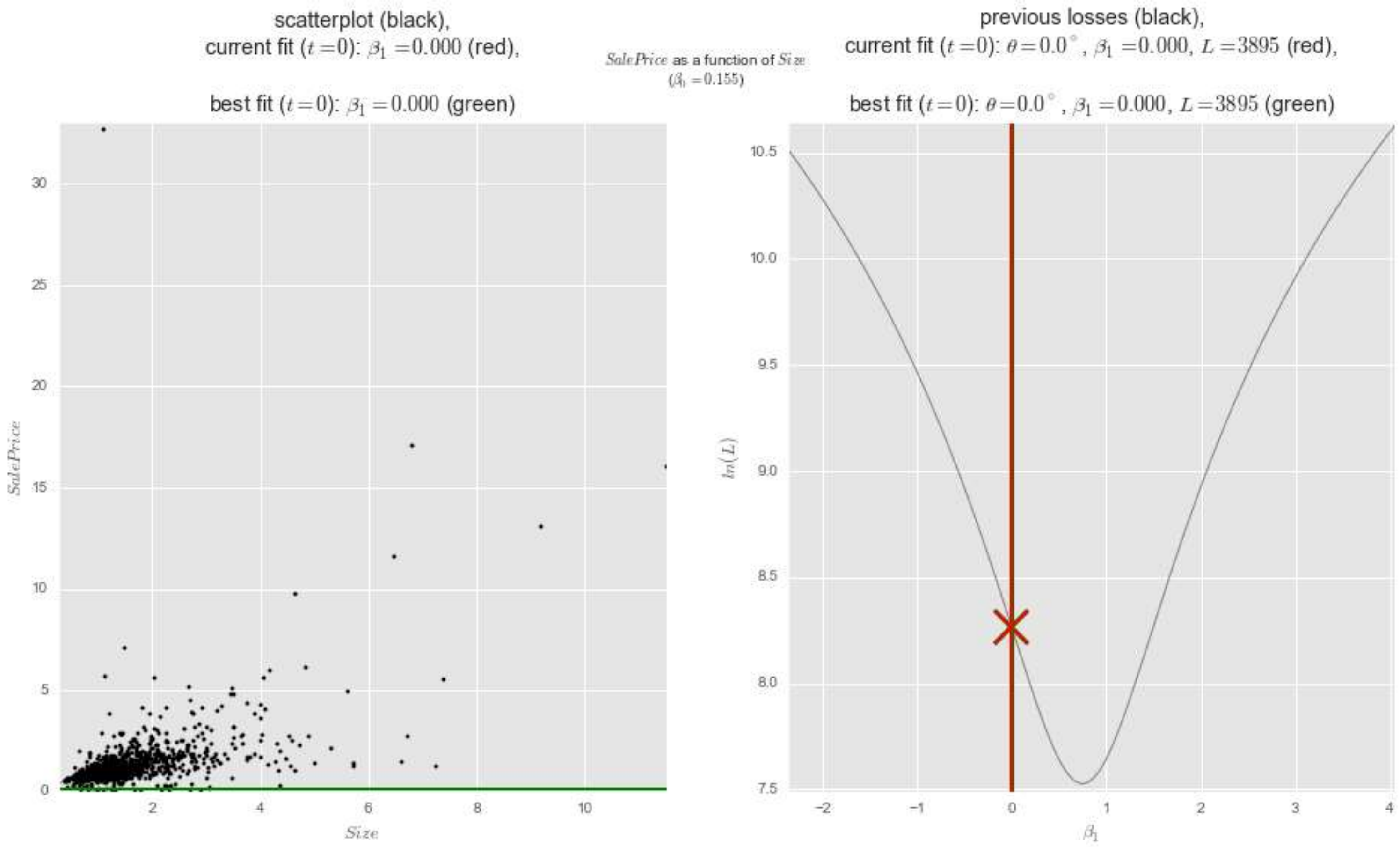
Demo | $t = 25$


scatterplot (black),
best fit ($t=25$): $\beta_1 = 0.775$ (green)
$SalePrice$ as a function of $Size$
($\beta_0 = 0.155$)
previous losses (black),
best fit ($t=25$): $\theta = 37.8^\circ$, $\beta_1 = 0.775$, $L = 1861$ (green)
$SalePrice$
$Size$
0
5
10
15
20
25
30
2
4
6
8
10
$ln(L)$
β_1
7.5
7.6
7.7
7.8
7.9
8.0
8.1
8.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4

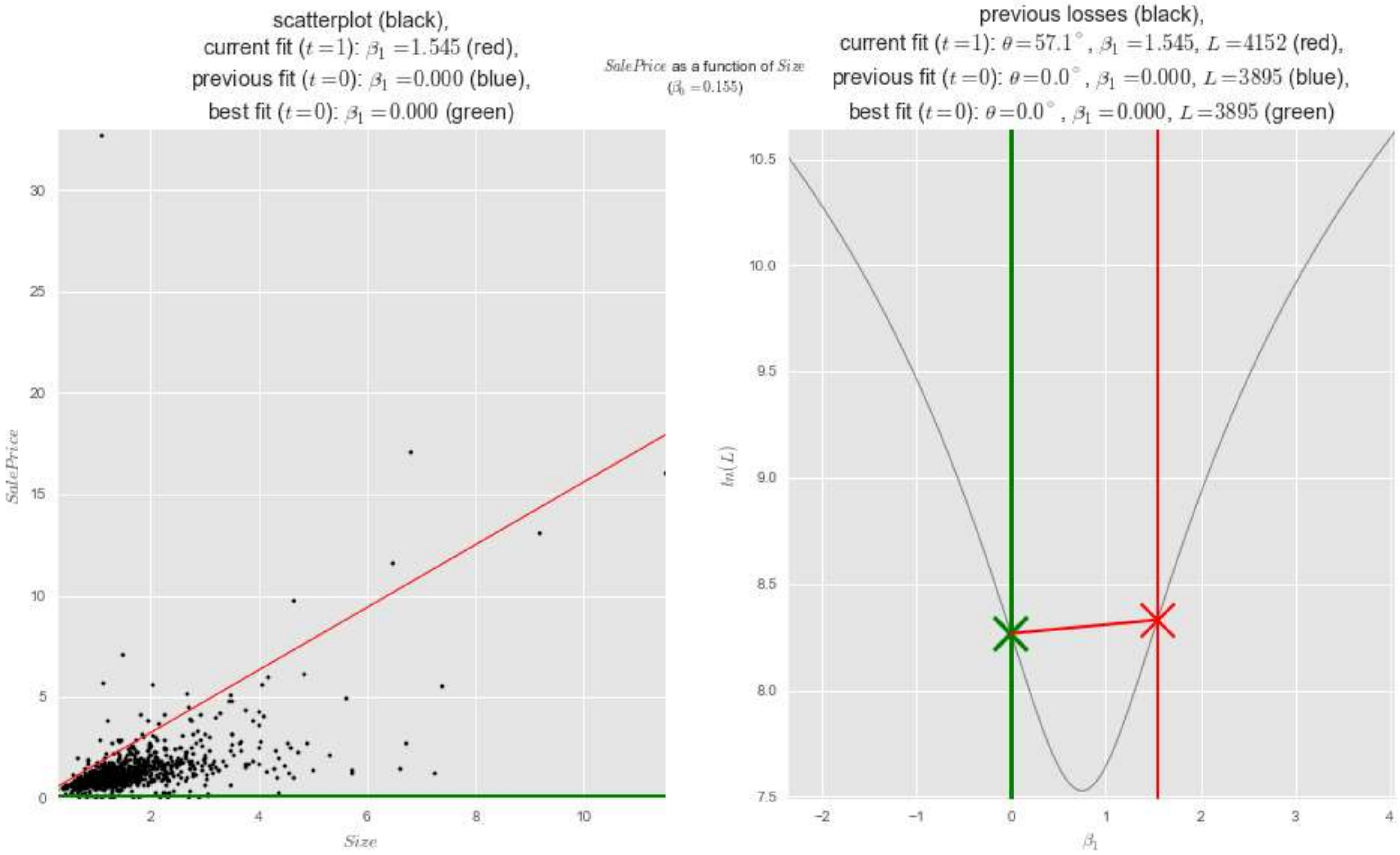
Demo | Again with $\alpha = .55$…



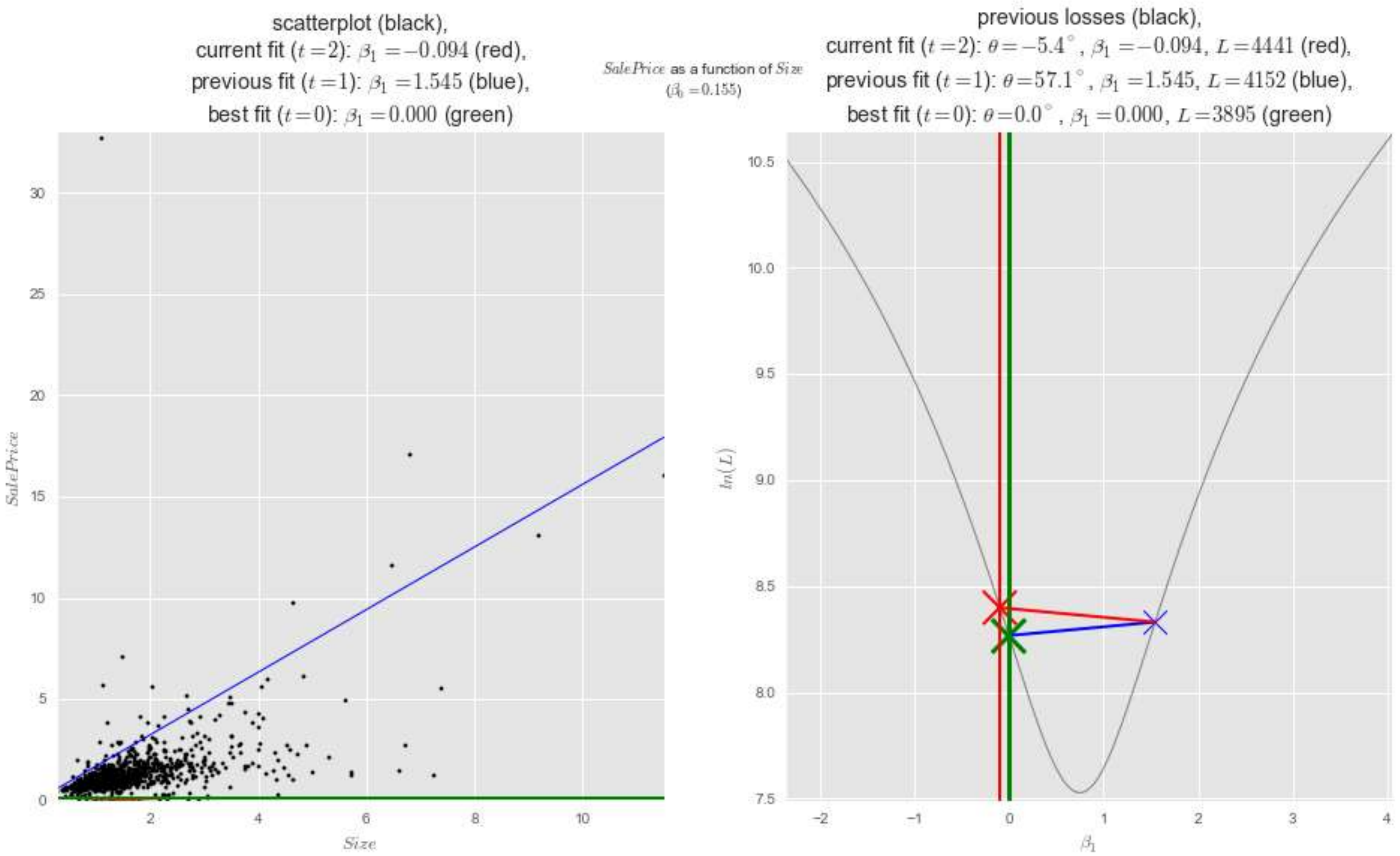
Demo | $t = 0$


scatterplot (black),
current fit ($t=0$): $\beta_1 = 0.000$ (red),
best fit ($t=0$): $\beta_1 = 0.000$ (green)
$SalePrice$ as a function of $Size$
($\beta_1 = 0.155$)
previous losses (black),
current fit ($t=0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (red),
best fit ($t=0$): $\theta = 0.0^\circ$, $\beta_1 = 0.000$, $L = 3895$ (green)
$SalePrice$
30
25
20
15
10
5
0
2
4
6
8
10
$Size$
$ln(L)$
10.5
10.0
9.5
9.0
8.5
8.0
7.5
−2
−1
0
1
2
3
4
β_1

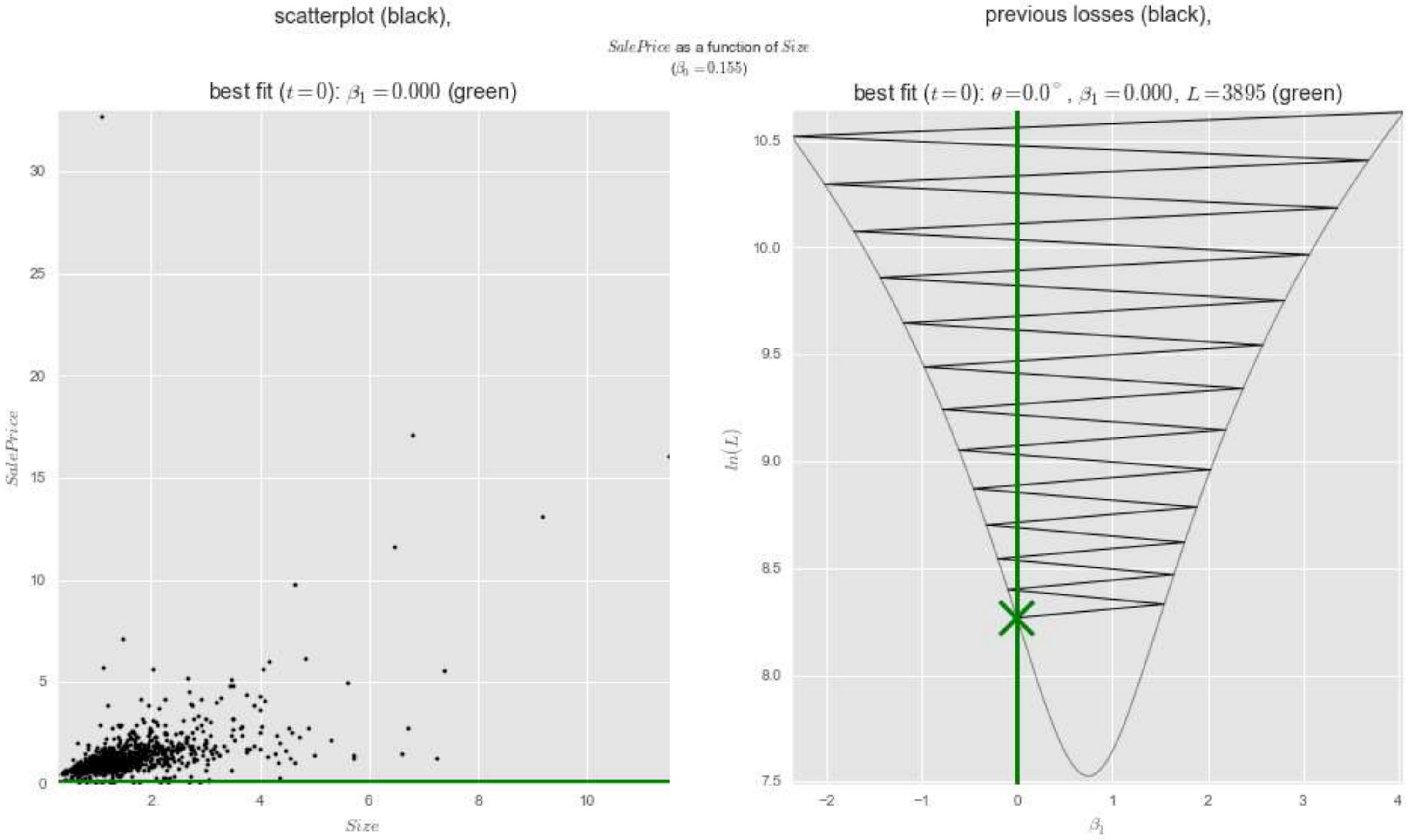
Demo | $t = 1$ | The gradient descent “overshoot”



Demo | $t = 2$ | We overshoot again but now it seems that we aren’t converging…



Demo | $t = 25$


scatterplot (black),
SalePrice as a function of Size
(β0 = 0.155)
best fit (t = 0): β1 = 0.000 (green)
SalePrice
Size
0
5
10
15
20
25
30
2
4
6
8
10
previous losses (black),
best fit (t = 0): θ = 0.0°, β1 = 0.000, L = 3895 (green)
ln(L)
β1
7.5
8.0
8.5
9.0
9.5
10.0
10.5
−2
−1
0
1
2
3
4

We can generalize the gradient function for more than one variable as long as the function to optimize is convex

- Goal

$$\min_{\beta_0,\beta_1} L(\beta_0, \beta_1)$$

(i.e., minimizing the least squares)

- Gradient Descent Algorithm
  - Start with some β_0 and β_1
  - Repeat until convergence

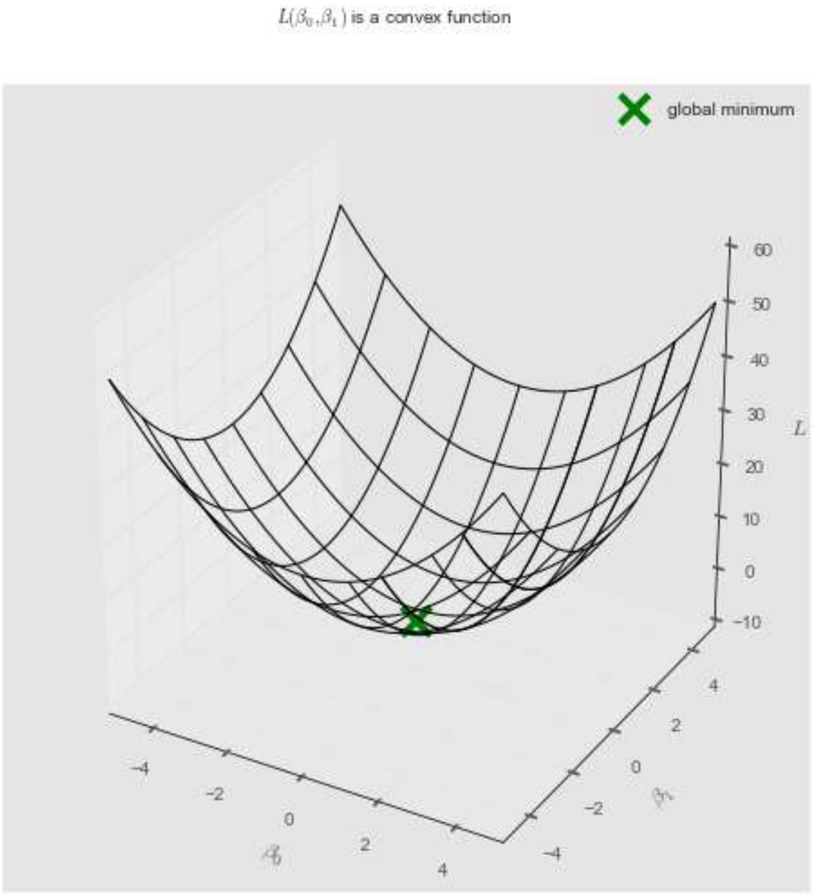
$$\beta_0 \coloneqq \beta_0 - \alpha \underbrace{\frac{\partial}{\partial \beta_0} L(\beta_0, \beta_1)}_{\frac{1}{m}\sum_{i=1}^{m} 1\cdot\underbrace{(\beta_0+\beta_1\cdot x_i-y_i)}_{-\varepsilon_i}} = \beta_0 + \frac{\alpha}{m}\sum_{i=1}^{m} \varepsilon_i$$

$$\beta_1 \coloneqq \beta_1 - \alpha \underbrace{\frac{\partial}{\partial \beta_1} L(\beta_0, \beta_1)}_{\frac{1}{m}\sum_{i=1}^{m} x_i\cdot\underbrace{(\beta_0+\beta_1\cdot x_i-y_i)}_{-\varepsilon_i}} = \beta_1 + \frac{\alpha}{m}\sum_{i=1}^{m} \varepsilon_i \cdot x_i$$

Here’s some code to estimate $\hat{\beta}_0$ and $\hat{\beta}_1$. Note that the δs need to be all computed together, i.e., before updating $\hat{\beta}_0$ or $\hat{\beta}_1$

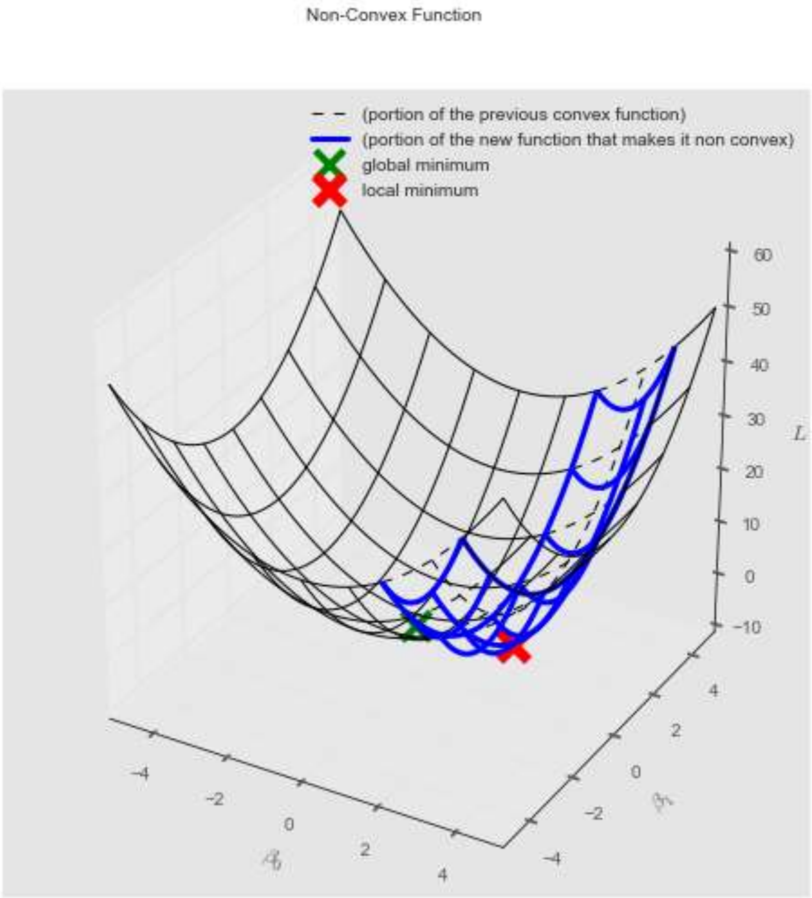
```
beta_0 = 0
beta_1 = 0

for _ in range(n):
    beta_0_delta = alpha *\
        (y - y_hat(beta_0, beta_1, x)).mean()
    beta_1_delta = alpha *\
        ((y - y_hat(beta_0, beta_1, x)) * x).mean()

    beta_0 += beta_0_delta
    beta_1 += beta_1_delta
```

$L(\beta_0, \beta_1)$ is a convex function so we can use gradient descent to estimate $\hat{\beta}_0$ and $\hat{\beta}_1$

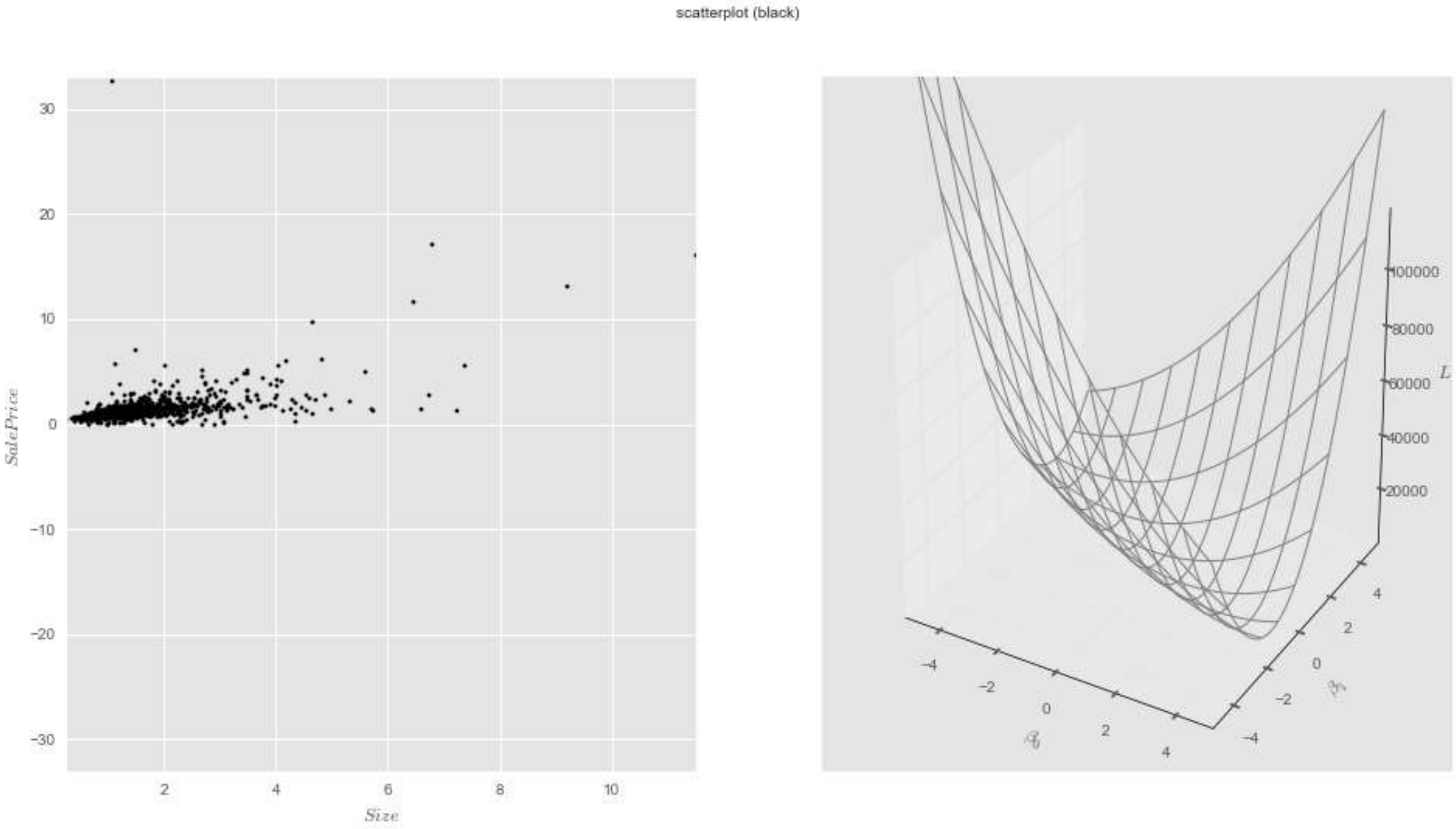


$L(\beta_0, \beta_1)$ is a convex function

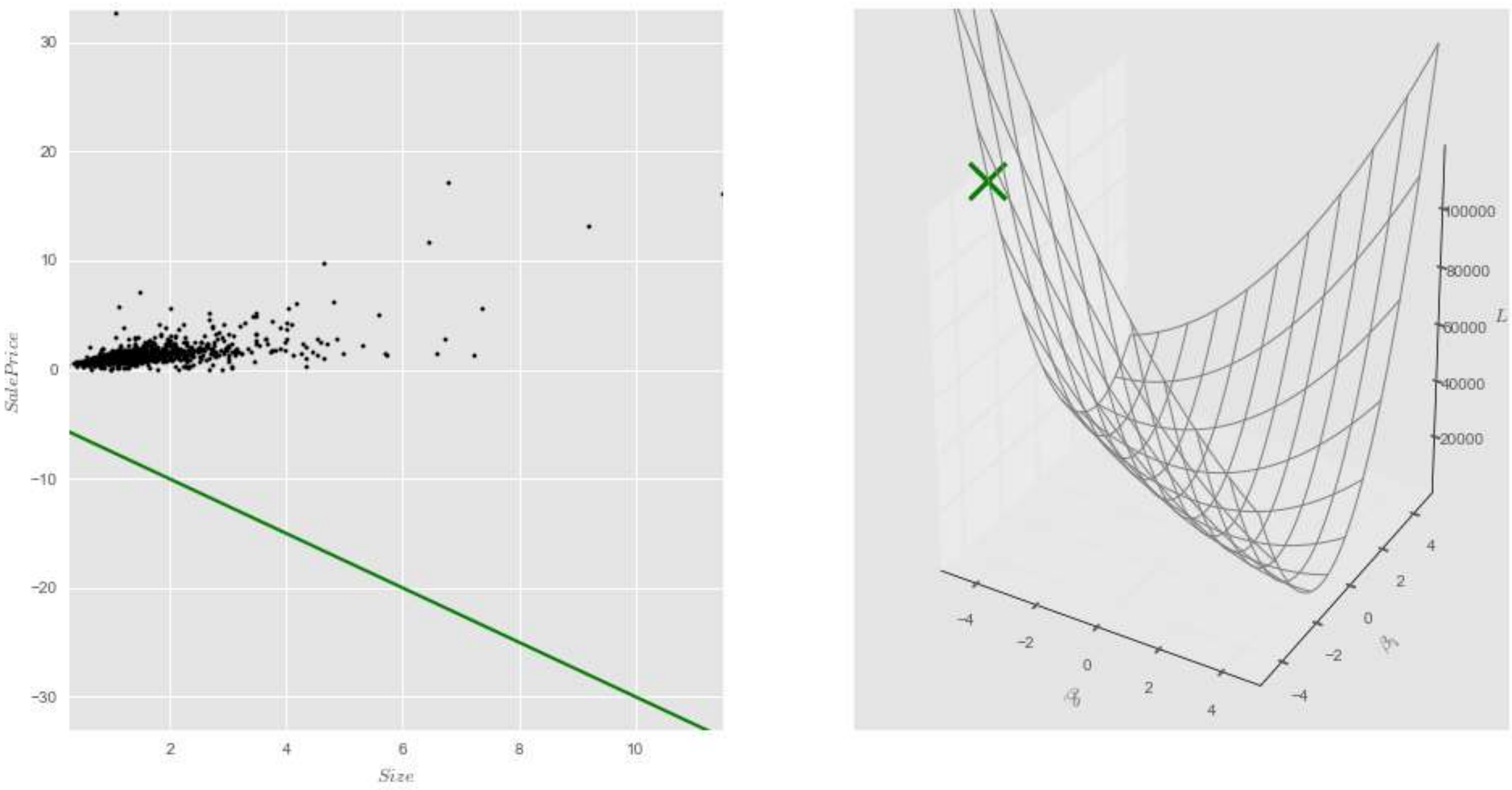
On the other hand, here’s a function that is not convex



Demo | Our familiar graphs but with $L(\beta_0, \beta_1)$ on the right instead of $L(\beta_1)$



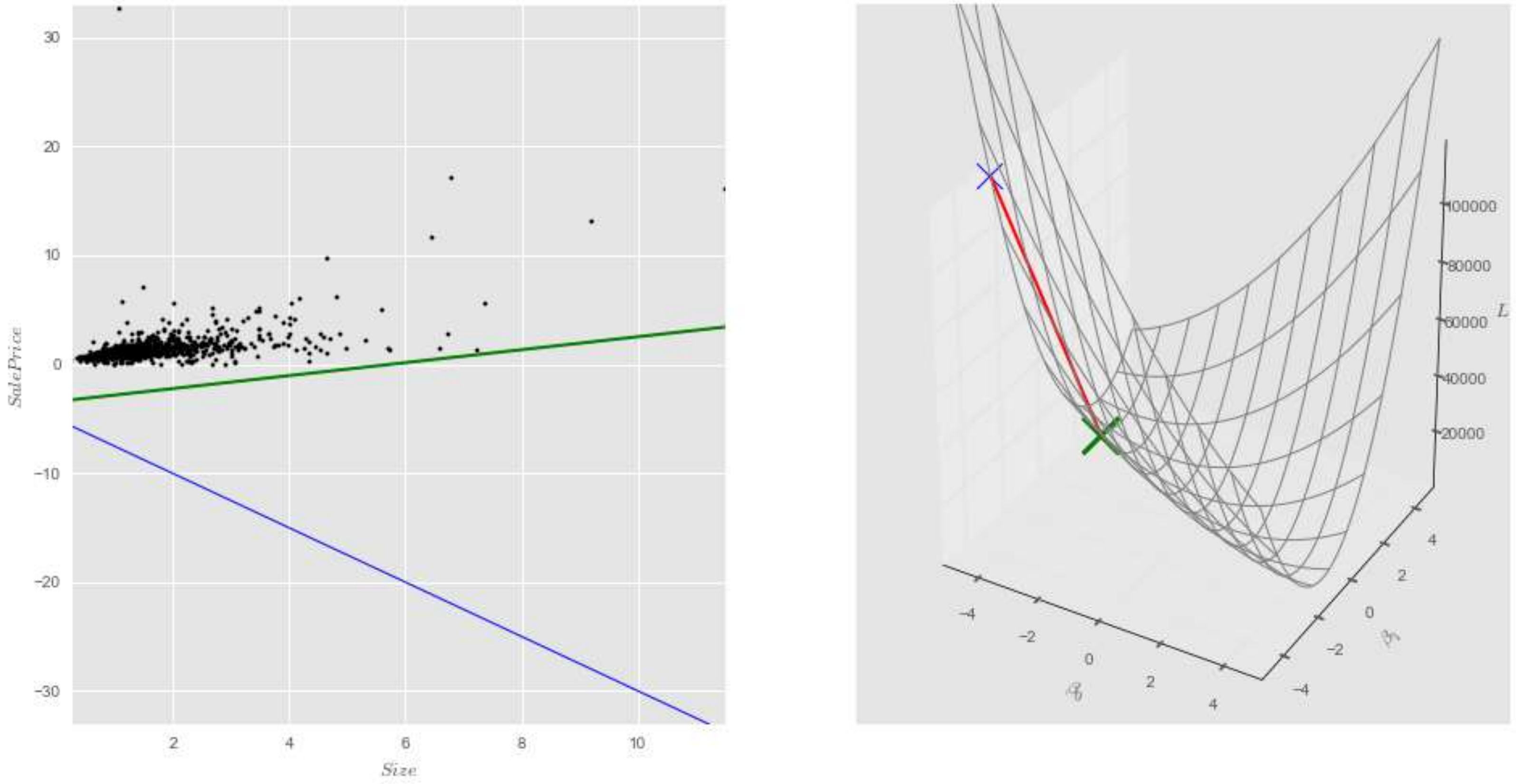
Demo | $t = 0$ ($\hat{\beta}_0$ arbitrarily set to -5, $\hat{\beta}_1$ to -2.5)



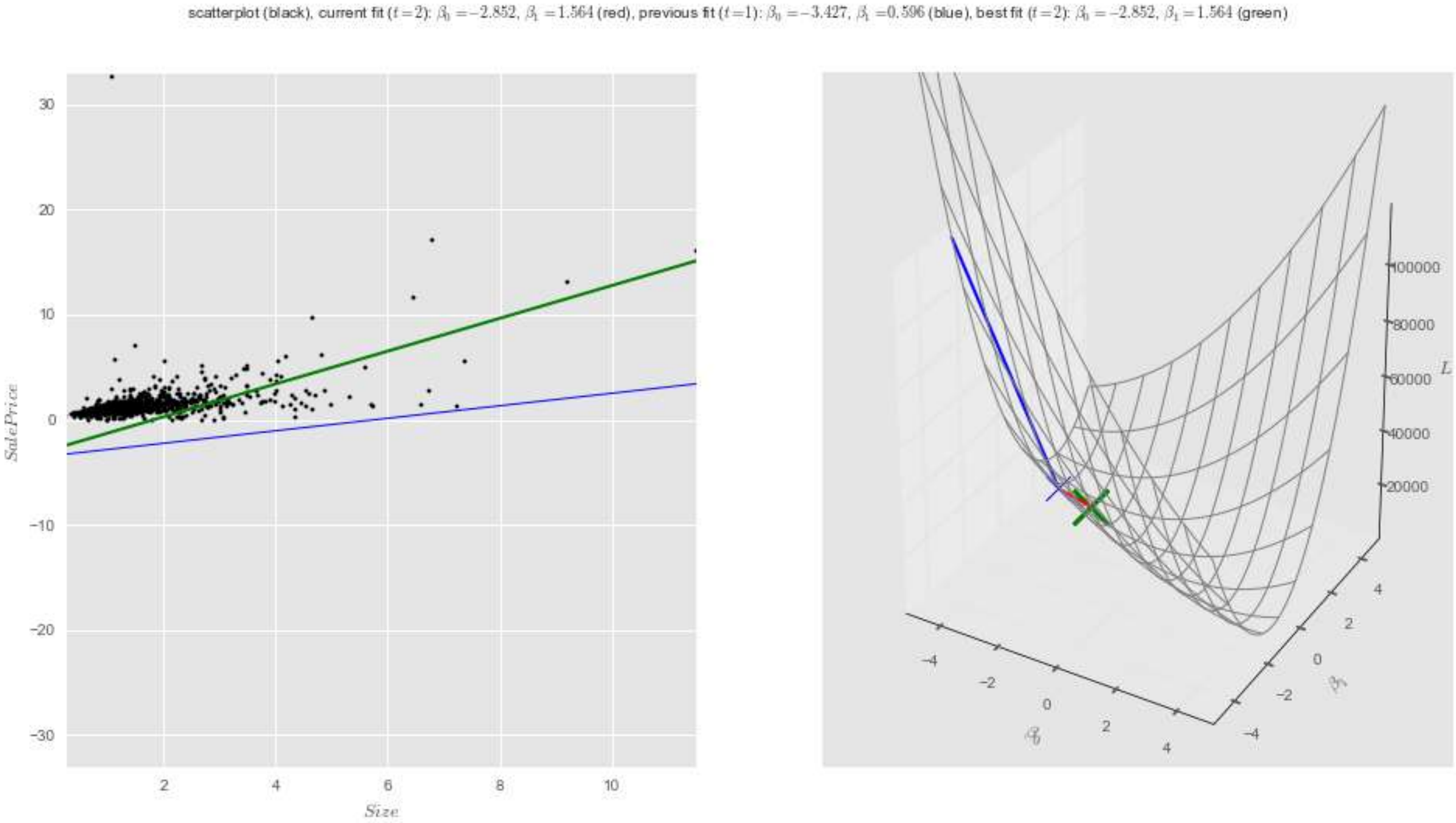
scatterplot (black), current fit ($t=0$): $\beta_0 = -5.000$, $\beta_1 = -2.500$ (red), best fit ($t=0$): $\beta_0 = -5.000$, $\beta_1 = -2.500$ (green)

Demo | $t = 1$ | The gradient descent made a "big jump" and updated both $\hat{\beta}_0$ and $\hat{\beta}_1$

scatterplot (black), current fit ($t=1$): $\beta_0 = -3.427$, $\beta_1 = 0.596$ (red), previous fit ($t=0$): $\beta_0 = -5.000$, $\beta_1 = -2.500$ (blue), best fit ($t=1$): $\beta_0 = -3.427$, $\beta_1 = 0.596$ (green)

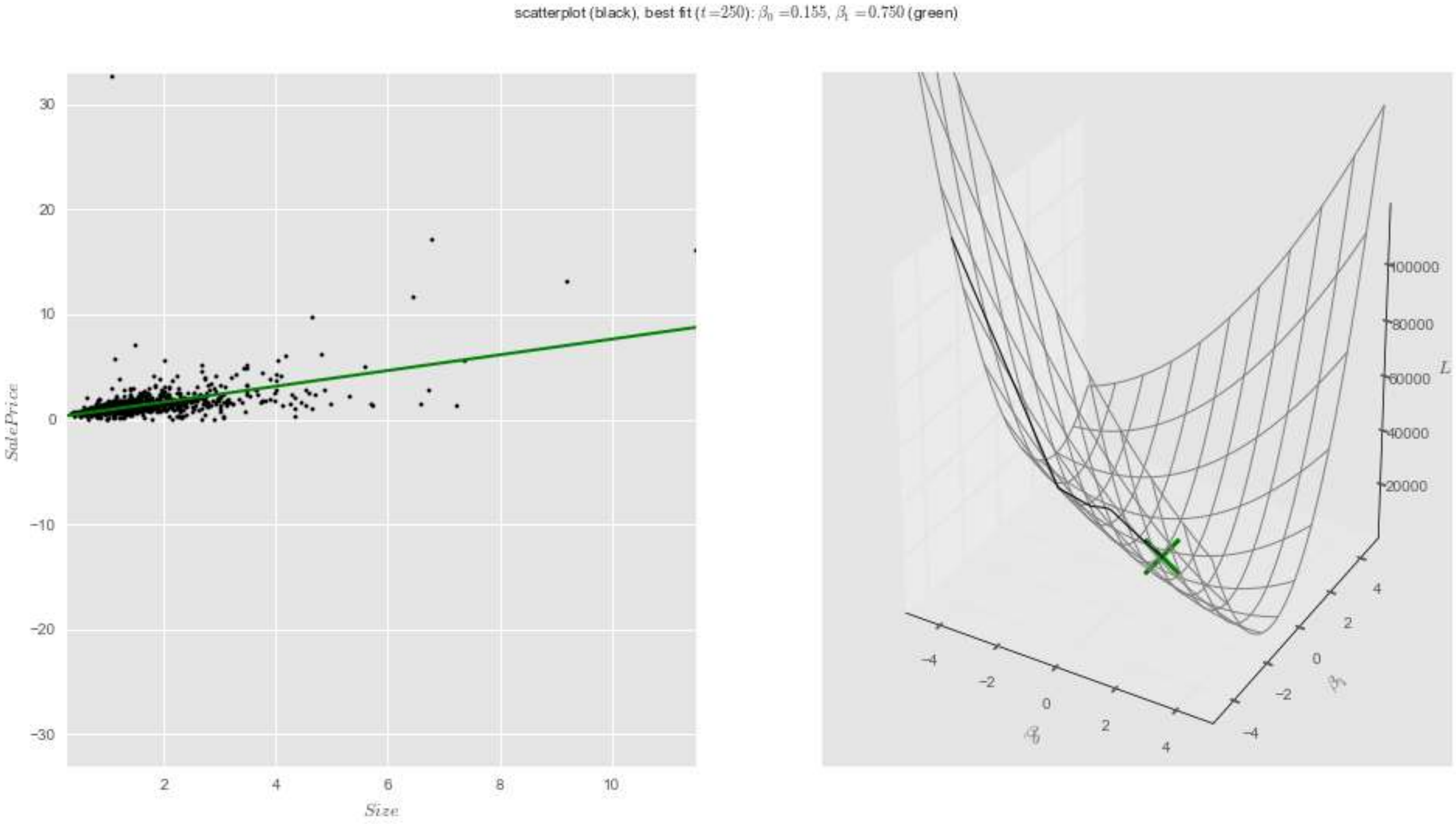


Demo | $t = 2$



scatterplot (black), current fit ($t=2$): $\beta_0=-2.852$, $\beta_1=1.564$ (red), previous fit ($t=1$): $\beta_0=-3.427$, $\beta_1=0.596$ (blue), best fit ($t=2$): $\beta_0=-2.852$, $\beta_1=1.564$ (green)

Demo | On the left, the best fitted line with $\hat{\beta}_0 = .155$ and $\hat{\beta}_1 = .750$. On the right the points of $L(\beta_0, \beta_1)$ following the gradient descent



scatterplot (black), best fit ($t = 250$): $\beta_0 = 0.155$, $\beta_1 = 0.750$ (green)

DS

Simple Linear Regression

Common Regression Assumptions

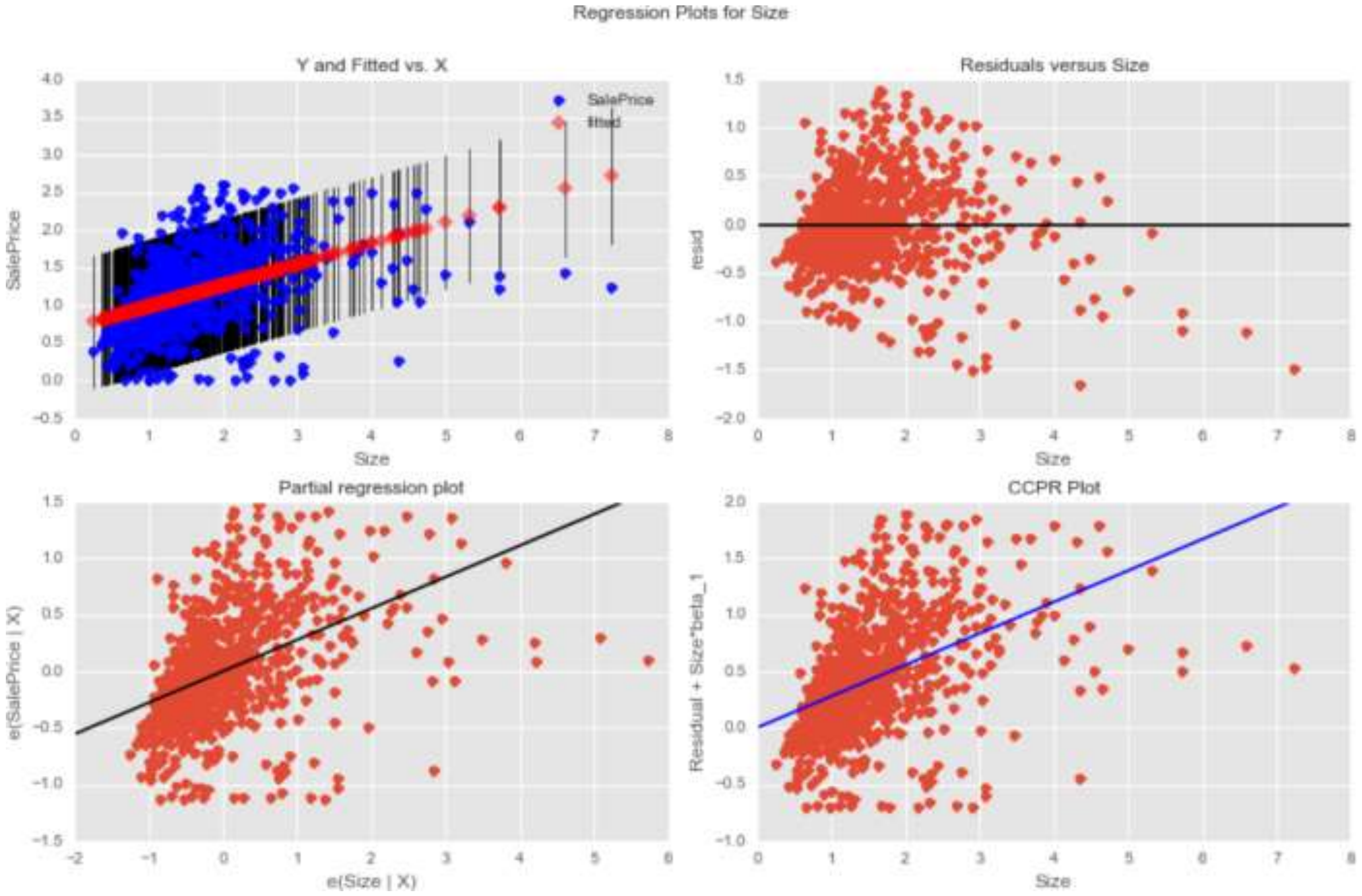
Common Regression Assumptions (part 1)

- The model is linear
  - x significantly explains y
- $\varepsilon \sim N(0,\ \cdot)$
  - Specifically, we expect ε to be 0 on average: $\mu_\varepsilon = 0$
- x and ε are independent
  - $\rho(x, \varepsilon) = 0$

Simple Linear Regression

Codealong – Part B
How to check modeling assumptions?

.plot_regress_exog()



Simple Linear Regression

How to check modeling assumptions?

`.plot_regress_exog()` to check modeling assumptions with respect to a single regressor

- Scatterplot of observed values (y) compared to fitted values ($\hat{y}$) with confidence intervals against the regressor (x)
- `.plot_fit()`

- "Residual Plot"
- Scatterplot of the model's residuals ($\hat{\varepsilon}$) against the regressor (x)

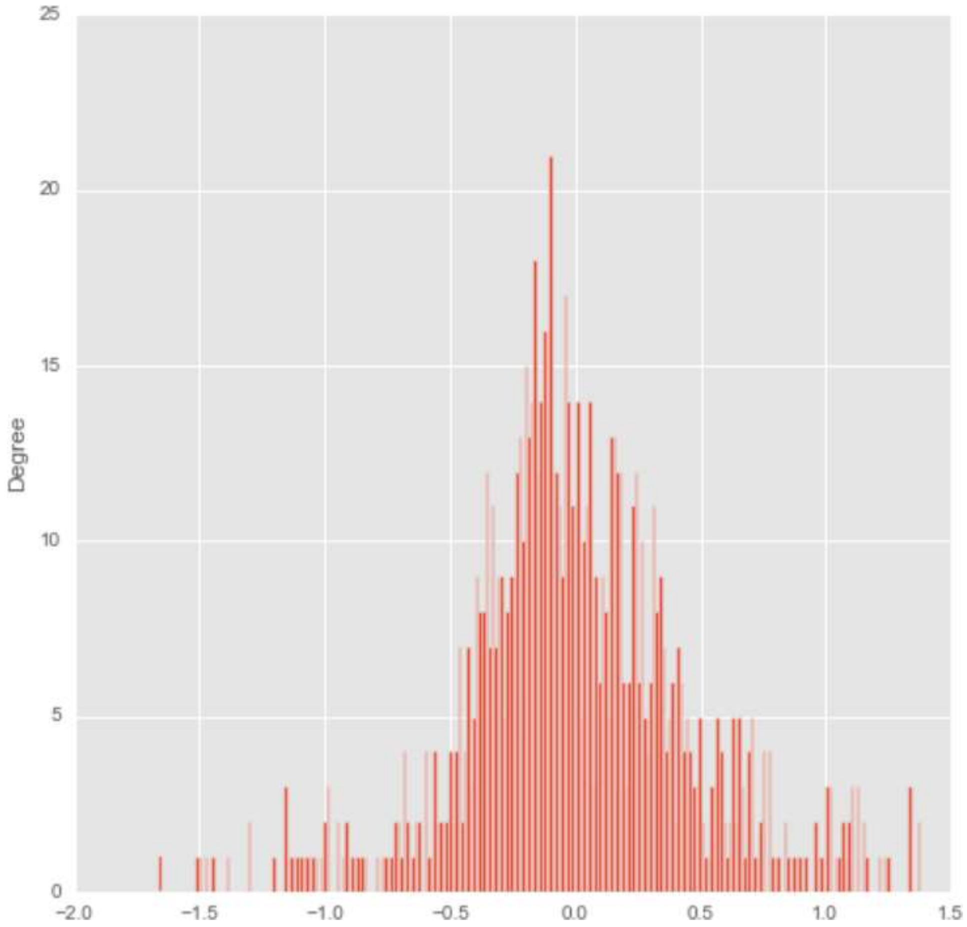
- "Partial Regression Plot" and "CCPR Plot (Component and Component-Plus-Residual)"
  - (useful for multiple regression) (more on this later)

DS

Simple Linear Regression

Codealong – Part C1
How to check normality assumption?

Is this normally distributed?



Simple Linear Regression

How to check normality assumption?

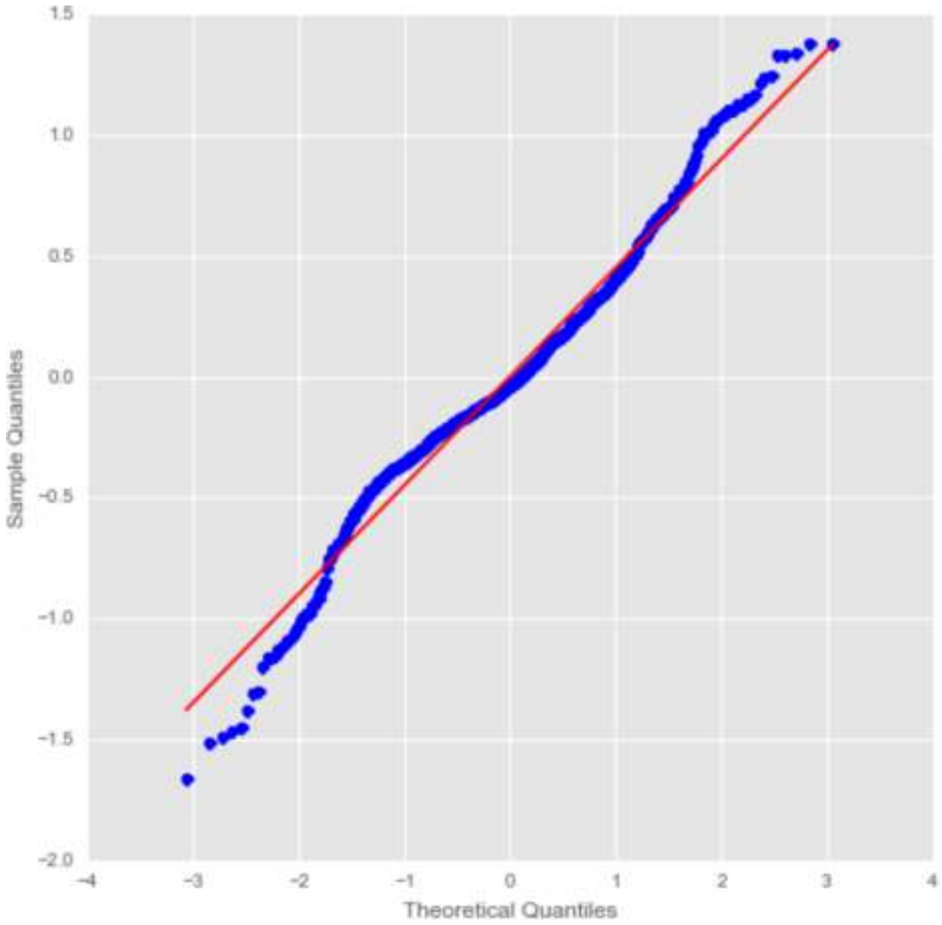
`.qqplot()` to check normality assumption

- "Quantile-Quantile (q-q) Plot"
- Graphical technique for determining if two datasets come from populations with a common distribution
- Plot of the quantiles of the first dataset (vertically) against the quantiles of the second's (horizontally)
- If unspecified, the second dataset will default to $N(0, 1)$
- If the two datasets come from a population with the same distribution, the points should fall approximately along a 45-degree reference line
- The greater the departure from this reference line, the greater the evidence for the conclusion that the datasets have come from populations with different distributions

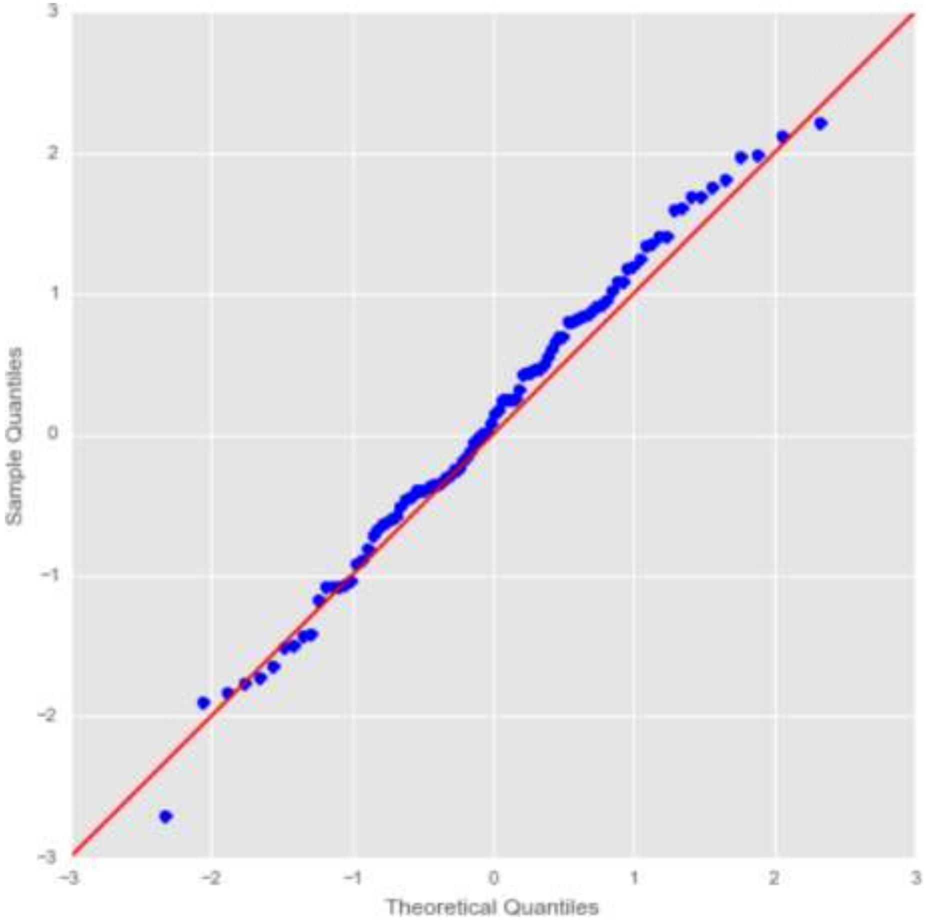
Simple Linear Regression

Codealong – Part C2
How to check normality assumption?

`.qqplot()` with `line = ‘s’`

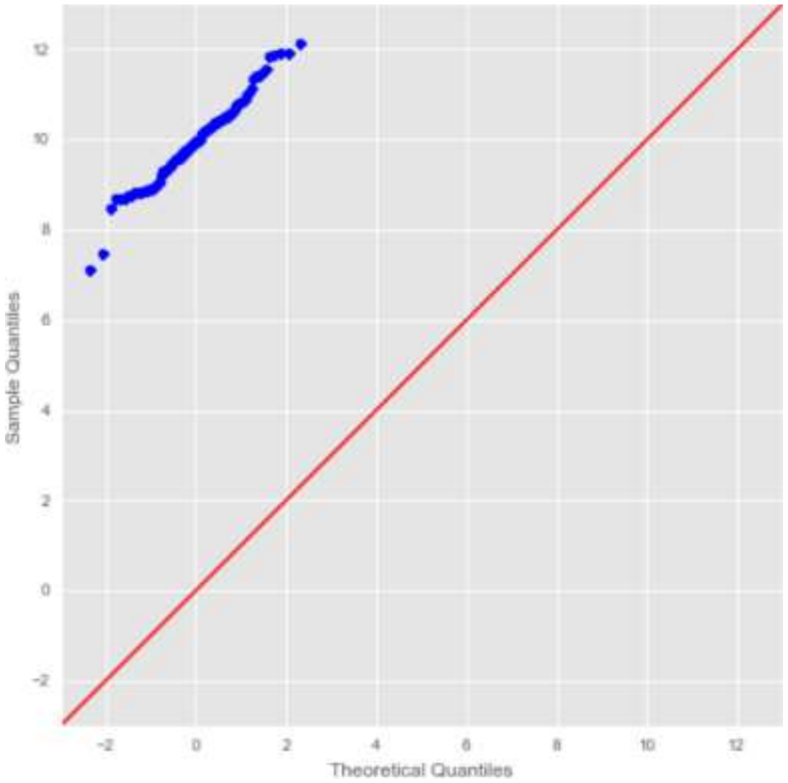


`.qqplot()` with `line = ‘45’`; $N(0,1)$ vs. $N(0,1)$

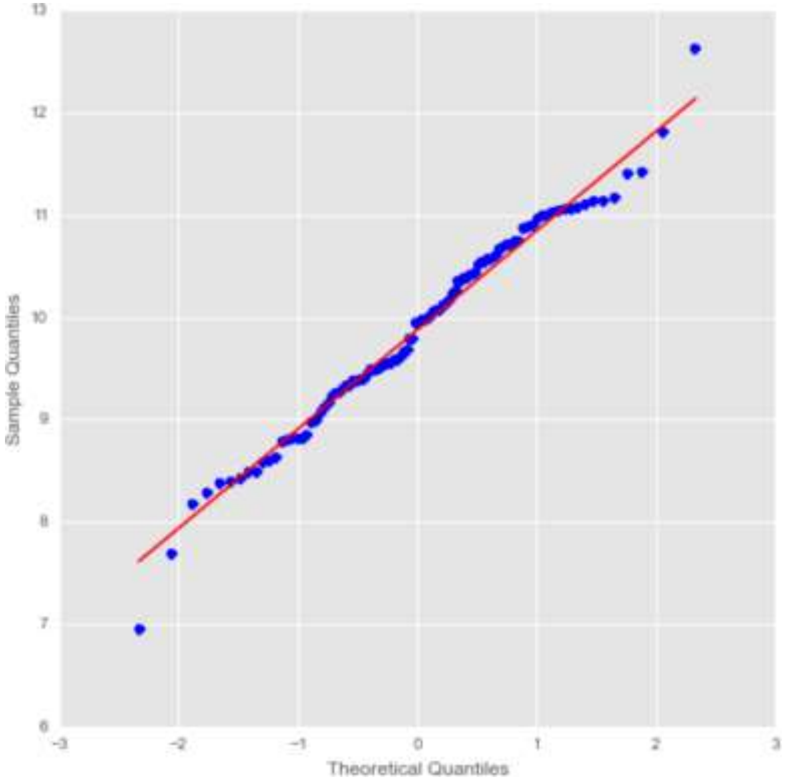


`.qqplot()`; $N(10, 1)$ vs. $N(0, 1)$

`line = ‘45’`

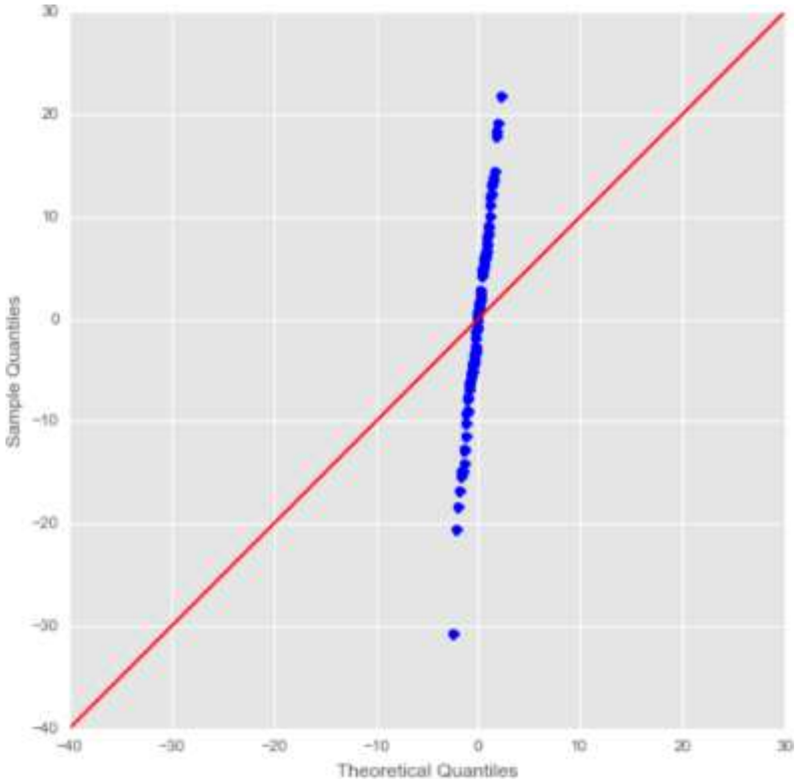


`line = ‘s’`



`.qqplot()`; $N(0, 10)$ vs. $N(0, 1)$

`line = ‘45’`



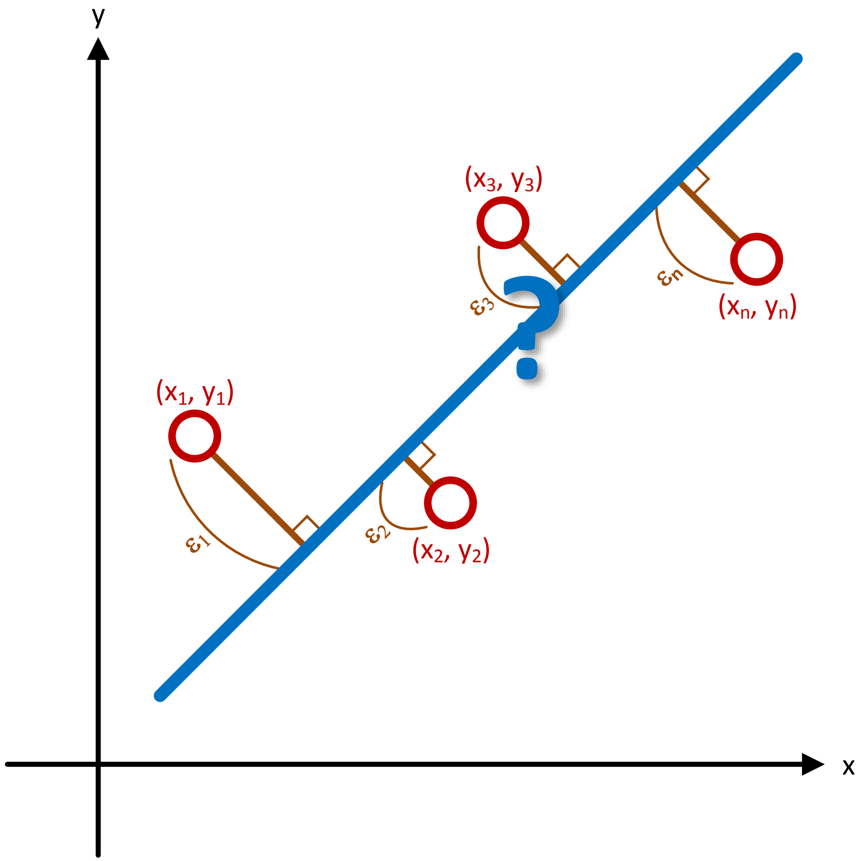
`line = ‘s’`

?

Simple Linear Regression

There are many ways to fit a line

There are many ways to fit a line

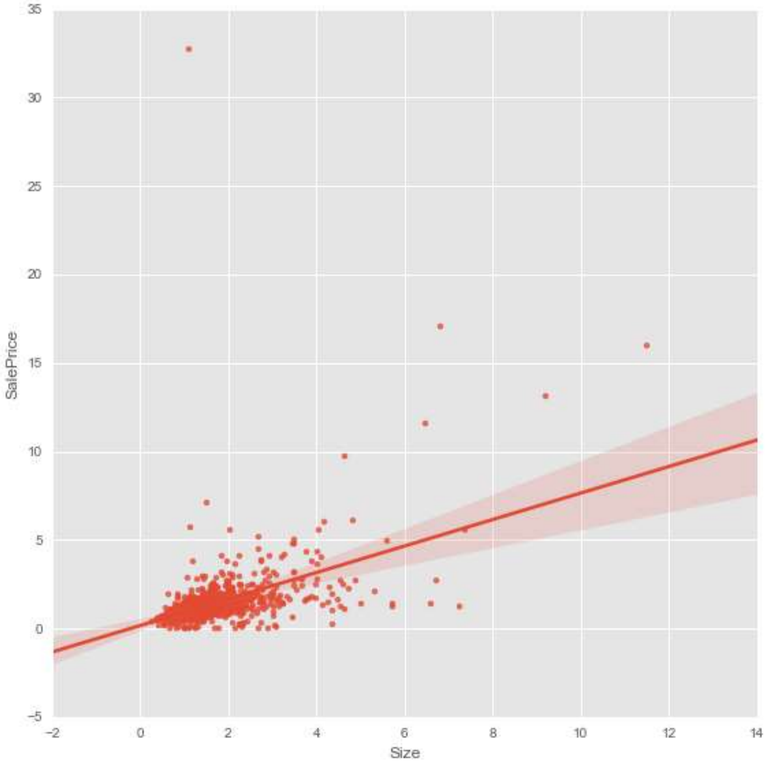


Simple Linear Regression

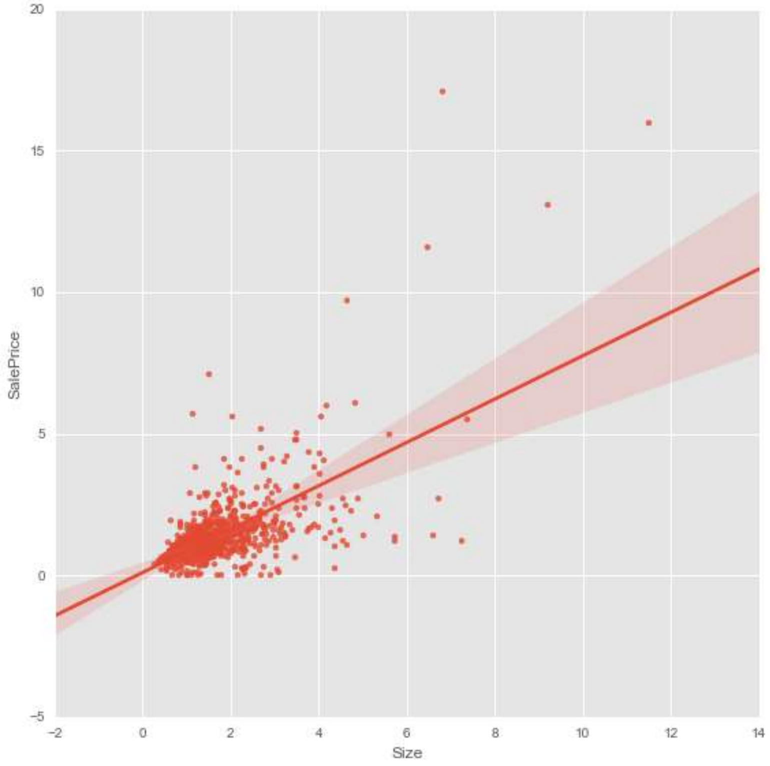
Codealong – Part D
Inference and Fit

Effect of outliers on linear regression modeling

All



"Top" Outlier Dropped

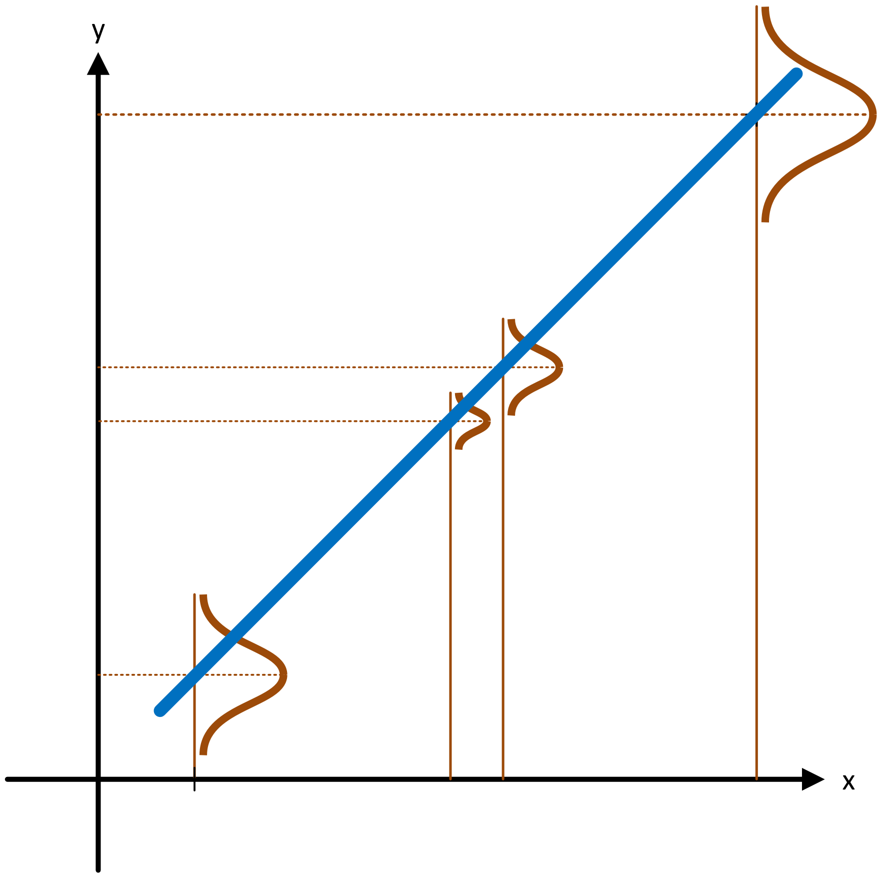


Simple Linear Regression

Inference, Fit, and R^2 (r-square)

Inference and Fit

- The deviations of the data from the best fitting line are normally distributed about the line. Since $\mu_\varepsilon = 0$, we "expect" that on average, the line will be correct
- How confident we are about how well the relationship holds depends on σ_ε^2



Measuring the fit of the line with R^2

- When a measure of how much of the total variation in y, $\sigma_y^2 = \beta^2\sigma_x^2 + \sigma_\varepsilon^2$, is explained by the portion associated with the explanatory variable x; also called systematic variation

$$R^2 = \rho_{xy}^2 = \frac{\beta^2\sigma_x^2}{\beta^2\sigma_x^2 + \sigma_\varepsilon^2}$$

- $0 \leq R^2 \leq 1$ (since $-1 \leq \rho_{xy} \leq 1$)

- $1 - R^2 = \frac{\sigma_\varepsilon^2}{\beta^2\sigma_x^2+\sigma_\varepsilon^2}$ is the idiosyncratic variation

R^2: Goodness of Fit

When $\mathbf{x}$ significantly explains $\mathbf{y}$	**When $\mathbf{x}$ does not significantly explains $\mathbf{y}$**
❑ The fit is **better**	❑ The fit is **worse**
❑ The **explained** systematic variation dominates	❑ The **unexplained** idiosyncratic variation dominates
❑ $\beta^2\sigma_x^2$ is high and/or σ_ε^2 is low	❑ $\beta^2\sigma_x^2$ is low and/or σ_ε^2 is high
❑ $\boldsymbol{R^2} = \frac{1}{1+\underbrace{\frac{\sigma_\varepsilon^2}{\beta^2\sigma_x^2}}_{\cong 0}}$ **is closer to 1**	❑ $\boldsymbol{R^2} = \frac{1}{1+\underbrace{\frac{\sigma_\varepsilon^2}{\beta^2\sigma_x^2}}_{\gg 1}}$ **is closer to 0**

Simple Linear Regression

Codealong – Part E

$\boldsymbol{R^2}$

Multiple Linear Regression

Multiple Linear Regression

- Simple linear regression with one variable can explain some variance, but using multiple variables can be much more powerful

- We can extend this model to several input variables, giving us the multiple linear regression model

$$y = \beta_0 + \beta_1 \cdot x_1 + \cdots + \beta_k \cdot x_k + \varepsilon$$

- Given $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n})$ and $y = (y_1, y_2, \ldots, y_n)$, we formulate the linear model as

$$y_i = \beta_0 + \beta_1 \cdot x_{1,i} + \cdots + \beta_k \cdot x_{k,i} + \varepsilon_i$$

- Given estimates for the model coefficients $\hat{\beta}_i$, we then predict y using

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_1 + \cdots + \hat{\beta}_k \cdot x_k$$

Multiple Linear Regression (cont.)

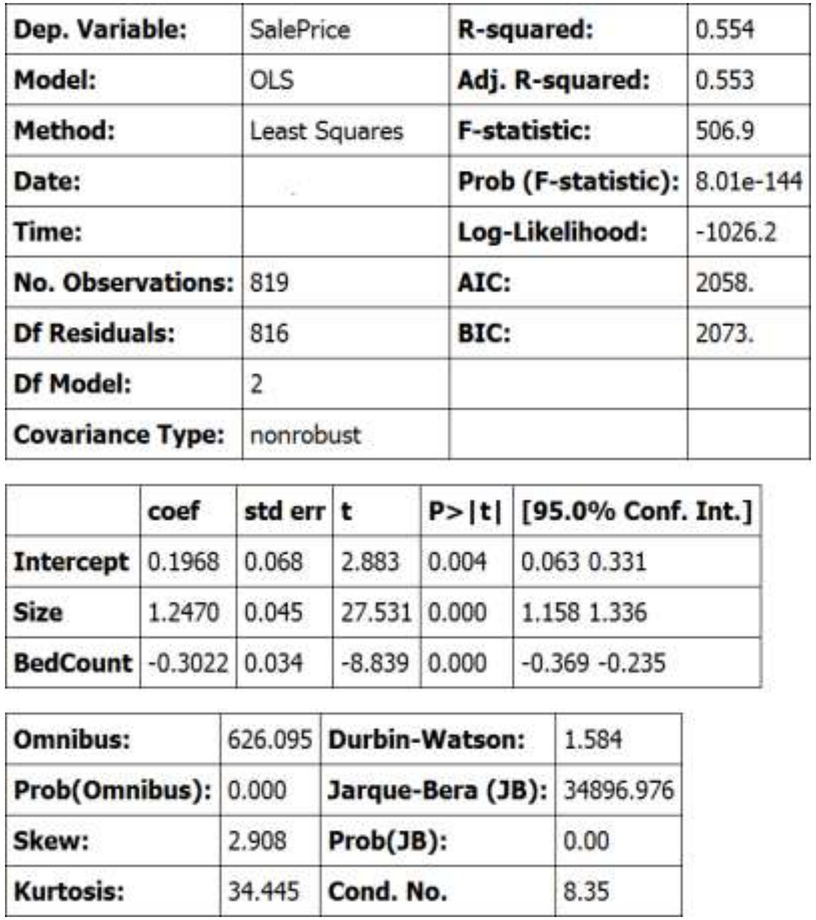
- E.g. (SF housing dataset),

$$\widehat{SalePrice} = \hat{\beta}_0 + \hat{\beta}_1 \cdot Size + \hat{\beta}_2 \cdot BedCount$$

Multiple Linear Regression

Codealong – Part F
Multiple Linear Regression

SalePrice ~ Size + BedCount (cont.)



Dep. Variable:	SalePrice	**R-squared:**	0.554
Model:	OLS	**Adj. R-squared:**	0.553
Method:	Least Squares	**F-statistic:**	506.9
Date:		**Prob (F-statistic):**	8.01e-144
Time:		**Log-Likelihood:**	-1026.2
No. Observations:	819	**AIC:**	2058.
Df Residuals:	816	**BIC:**	2073.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	0.1968	0.068	2.883	0.004	0.063 0.331
Size	1.2470	0.045	27.531	0.000	1.158 1.336
BedCount	-0.3022	0.034	-8.839	0.000	-0.369 -0.235

Omnibus:	626.095	**Durbin-Watson:**	1.584
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	34896.976
Skew:	2.908	**Prob(JB):**	0.00
Kurtosis:	34.445	**Cond. No.**	8.35

DS

Multiple Linear Regression

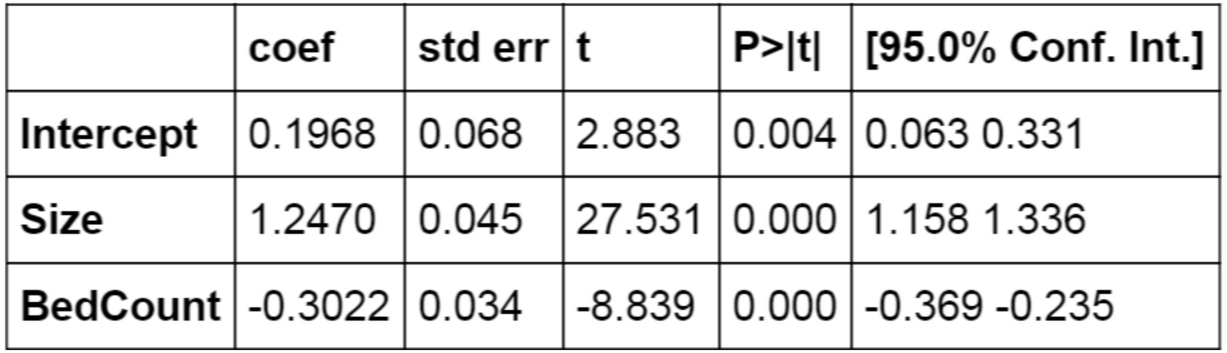
Activity | Knowledge Check

Activity | Knowledge Check

EXERCISE

DIRECTIONS (5 minutes)

1. Using the table below for `SalePrice ~ Size + BedCount`
   a. How do you interpret the model’s parameters? (units and values)



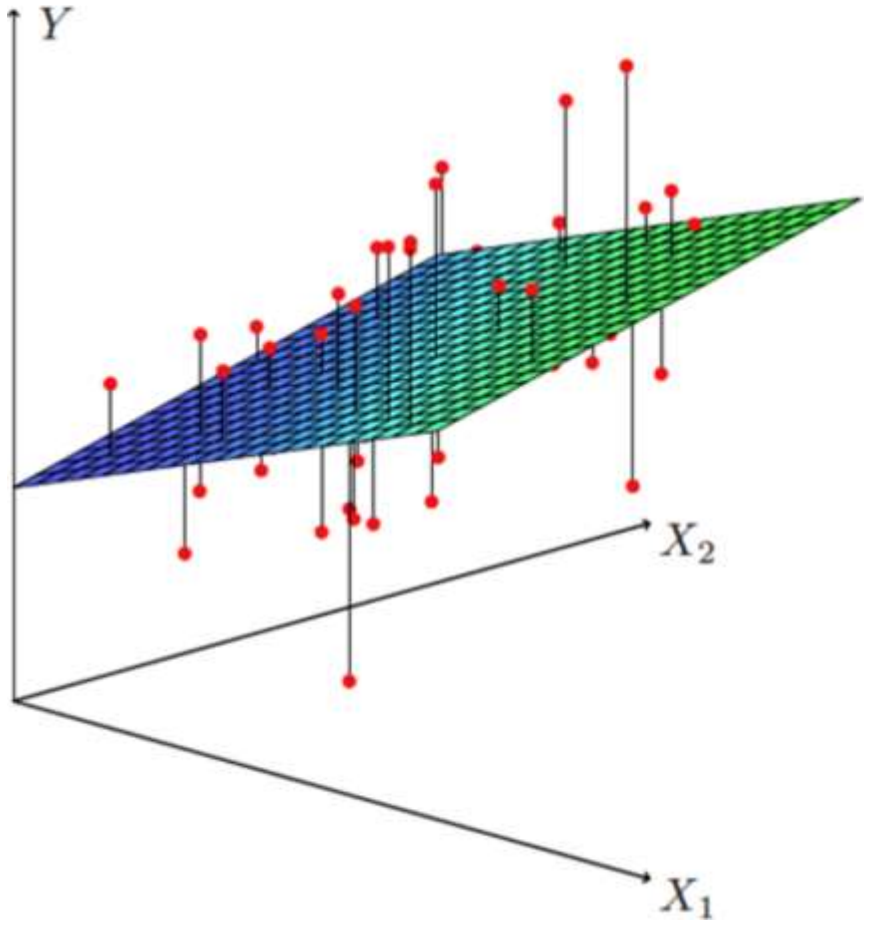
	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	0.1968	0.068	2.883	0.004	0.063 0.331
Size	1.2470	0.045	27.531	0.000	1.158 1.336
BedCount	-0.3022	0.034	-8.839	0.000	-0.369 -0.235

2. When finished, share your answers with your table

DELIVERABLE

Answers to the above questions

We can still estimate β_i with Ordinary Least Squares (OLS); here a fitted plane when $m = 2$



Multiple Linear Regression

Common Regression Assumptions (cont.)

Common Regression Assumptions (part 2)

- x_i are independent from each other (low multicollinearity)

- Multicollinearity (or collinearity) is a phenomenon in which two or more predictors in a multiple regression model are highly correlated, meaning that one can be linearly predicted from the others with a substantial degree of accuracy

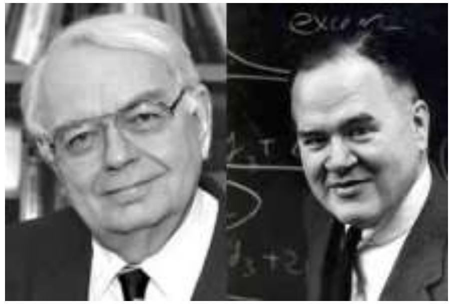
The ideal scenario: when predictors are uncorrelated

- Each coefficient can be estimated and tested separately
- β_i estimates the expected change in y per unit change in x_i, all other predictors held fixed
- However predictors usually change together

- Correlations amongst predictors cause problems
  - The variance of all coefficients tends to increase, sometimes dramatically
  - Interpretations become hazardous – when x_i changes, everything else changes

The woes of (interpreting) regression coefficients

- “The only way to find out what will happen when a complex system is distributed is to disturb the system, not merely to observe it passively” – Fred Mosteller and John Tukey



- “Essentially, all models are wrong, but some are useful” – George Box

Common Regression Assumptions (part 3)

- Linear regression also works best when
  - the data is normally distributed (it doesn't have to be)
  - (if data is not normally distributed, we could introduce *bias*)

DS

Multiple Linear Regression

Activity | Variable Transformations

Activity | Variable Transformations

EXERCISE

DIRECTIONS (5 minutes)

1. We want to run the following regression with the following non-linear terms:

$$\text{SalePrice} \sim Size^2 + \sqrt{BedCount}$$

   a. How can we linearize it?

2. When finished, share your answers with your table

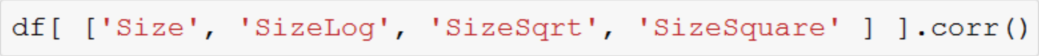
DELIVERABLE

Answers to the above questions

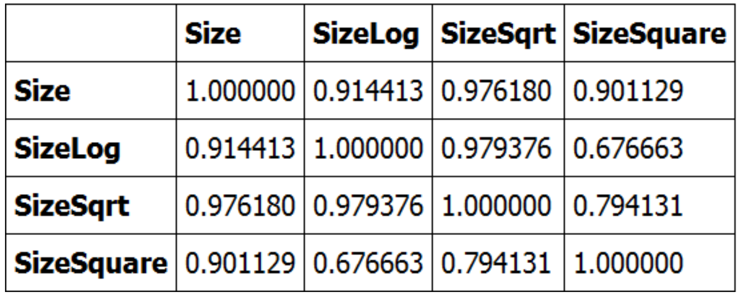
Multiple Linear Regression

Codealong – Part G
Variable Transformations (cont.)
Multicollinearity

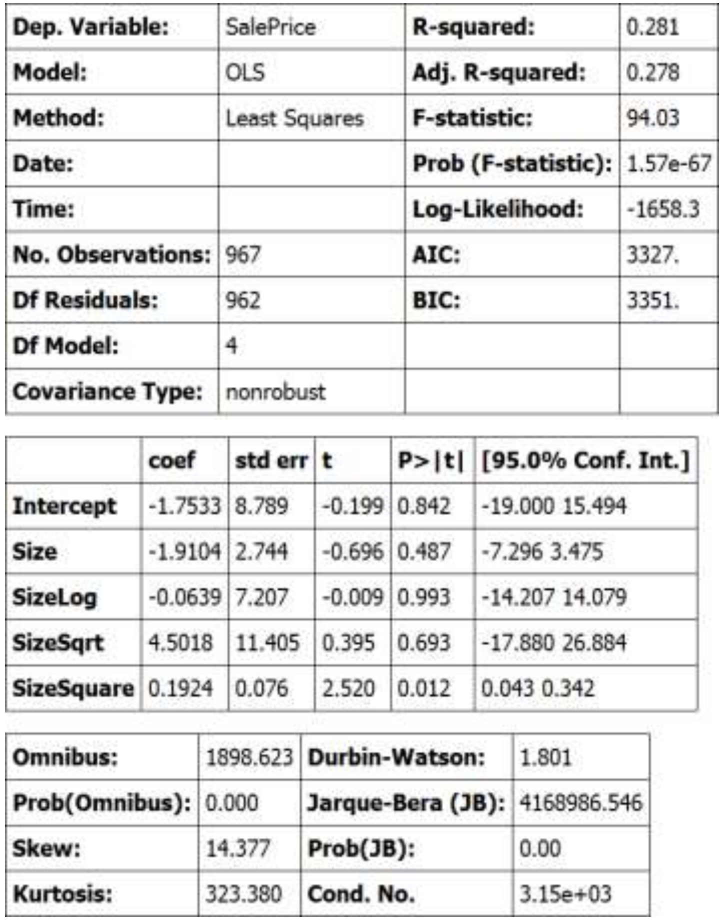
Multicollinearity between $Size$, $ln(Size)$, $\sqrt{Size}$, and $Size^2$. What happened?



```
df[ ['Size', 'SizeLog', 'SizeSqrt', 'SizeSquare' ] ].corr()
```



	Size	SizeLog	SizeSqrt	SizeSquare
Size	1.000000	0.914413	0.976180	0.901129
SizeLog	0.914413	1.000000	0.979376	0.676663
SizeSqrt	0.976180	0.979376	1.000000	0.794131
SizeSquare	0.901129	0.676663	0.794131	1.000000



Dep. Variable:	SalePrice	**R-squared:**	0.281
Model:	OLS	**Adj. R-squared:**	0.278
Method:	Least Squares	**F-statistic:**	94.03
Date:		**Prob (F-statistic):**	1.57e-67
Time:		**Log-Likelihood:**	-1658.3
No. Observations:	967	**AIC:**	3327.
Df Residuals:	962	**BIC:**	3351.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
Intercept	-1.7533	8.789	-0.199	0.842	-19.000 15.494
Size	-1.9104	2.744	-0.696	0.487	-7.296 3.475
SizeLog	-0.0639	7.207	-0.009	0.993	-14.207 14.079
SizeSqrt	4.5018	11.405	0.395	0.693	-17.880 26.884
SizeSquare	0.1924	0.076	2.520	0.012	0.043 0.342

Omnibus:	1898.623	**Durbin-Watson:**	1.801
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4168986.546
Skew:	14.377	**Prob(JB):**	0.00
Kurtosis:	323.380	**Cond. No.**	3.15e+03

`.plot_regress_exog()` (cont.)

- “Partial regression plot” (lower left)
- Partial regression for a single regressor
- The full model’s β_i is the fitted line’s slope
- The individual points can be used to assess the influence of points on the estimated coefficient
- `.plot_partregress()`

- “CCPR plot” (lower right)
  - Component and Component-Plus-Residual
- Refined partial residual plot
- Judge the effect of one regressor on the response variable by taking into account the effects of the other independent variables
- Scatterplot of the full model’s residuals ($\hat{\varepsilon}$) plus $\beta_i \cdot x_i$ against the regressor (x_i)
- `.plot_ccpr()`

Multiple Linear Regression

Codealong – Part H

$\bar{\boldsymbol{R}}^2$ *(Adjusted* $\boldsymbol{R}^2$ *)*

Multiple Linear Regression

$\boldsymbol{\bar{R}^2}$

$\bar{R}^2$

- R^2 increases as you add more variables in your model, even non-significant predictors; it's then tempting to add all the features from your dataset
- $\bar{R}^2$ attempts to adjust the explanatory power of regression models that contain different numbers of predictors so as to make comparisons possible

$$\bar{R}^2 = 1 - (1 - R^2) \cdot \frac{n-1}{n-k-1}$$

(n number of observations;

k number of parameters)

Lab

Introduction to Regression and Model Fit

Review

Linear regression is a simple approach to supervised learning. Pros and Cons are:

- Pros
  - Intuitive and well-understood
  - Can perform well with a small number of observations
  - Highly interpretable and simple to explain
  - Model training and prediction are fast
  - No need to standardize your data (features don't need scaling)
  - No tuning is required (excluding regularization which is a topic we won’t discuss)

- Cons
  - Assumes linear association among variables
  - Assumes normally distributed residuals
  - Outliers can easily affect coefficients

Review

You should now be able to:

- Define simple linear regression and multiple linear regression
- Build a linear regression model using a dataset that meets the linearity assumption
- Evaluate model fit
- Understand and identify multicollinearity in a multiple regression

Q & A

Next Class

Introduction to Regression and Model Fit, Part 2

Learning Objectives

After the next lesson, you should be able to:

- Define simple linear regression and multiple linear regression
- Build a linear regression model using a dataset that meets the linearity assumption
- Evaluate model fit
- Understand and identify multicollinearity in a multiple regression

Exit Ticket

Don’t forget to fill out your exit ticket here

Slides © 2016 Ivan Corneillet Where Applicable
Do Not Reproduce Without Permission