Linear Regression

15/02/2021

Overview

- Linear Regression Model
- Measuring error
- Error Surface
- Regression Algorithm
- Example

Prerequisite

- Derivative
- Finding Derivatives
- Chain Rule
- Partial Differentiation
- Dataset, Descriptive Features, Target

Supervised Learning

- In a supervised learning problem, you have access to input variables (X) and outputs (Y), and the goal is to predict an output given an input.

- Examples:
  - Housing prices (Regression): predict the price of a house based on features (size, location, age etc)
  - Cat vs. Dog (Classification): predict whether a picture is of a cat or a dog

Regression

- Predicting a continuous outcome variable:
  - Predicting a company's future stock price using its profit and other financial information
  - Retail - How much will be the daily, monthly, and yearly sales for a given store for the next three years?
  - How much will be the monthly electricity cost for the next three years?
  - How many customers will claim the insurance this year?
  - What will be the temperature for the next five days?
  - Predicting annual rainfall based on local flora and fauna

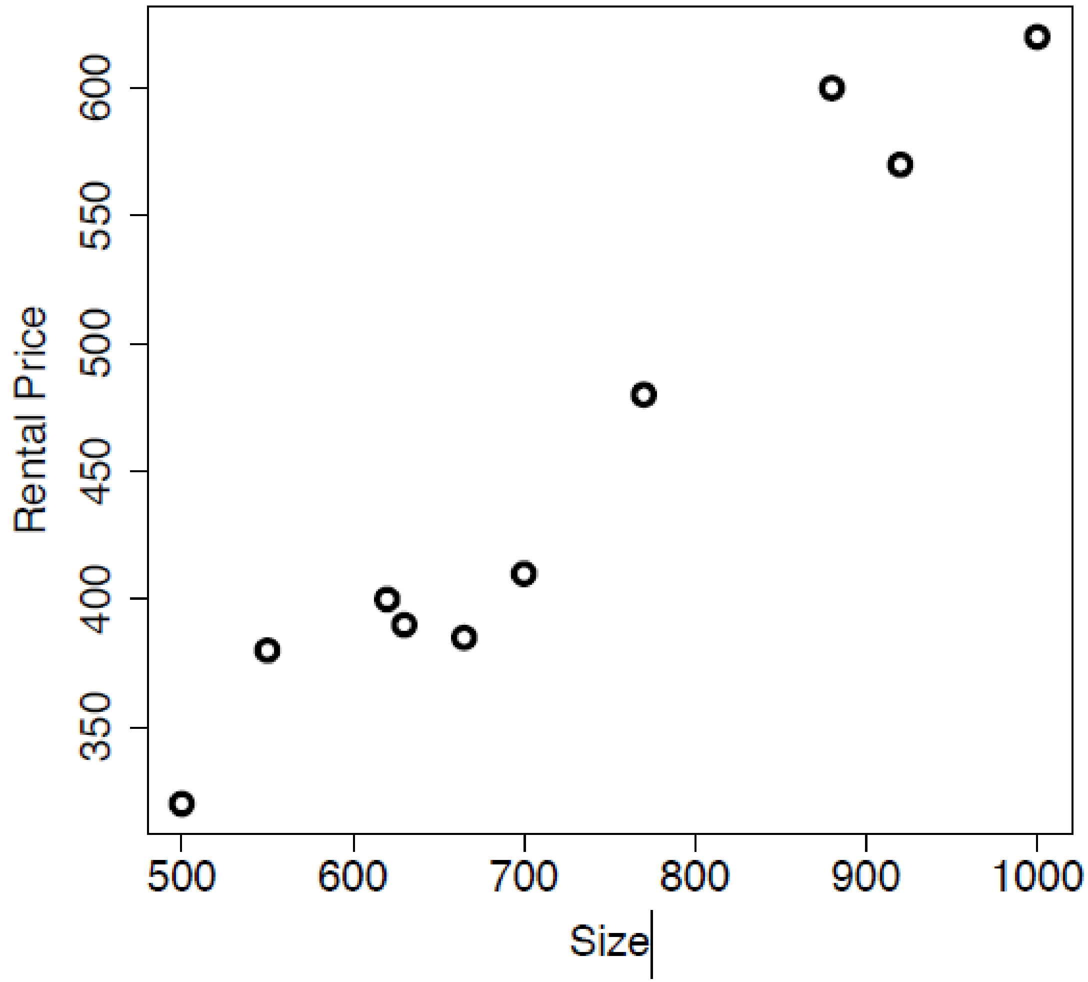
Ex: Predicting the rental price of an Office

A dataset that includes office rental prices and a number of descriptive features

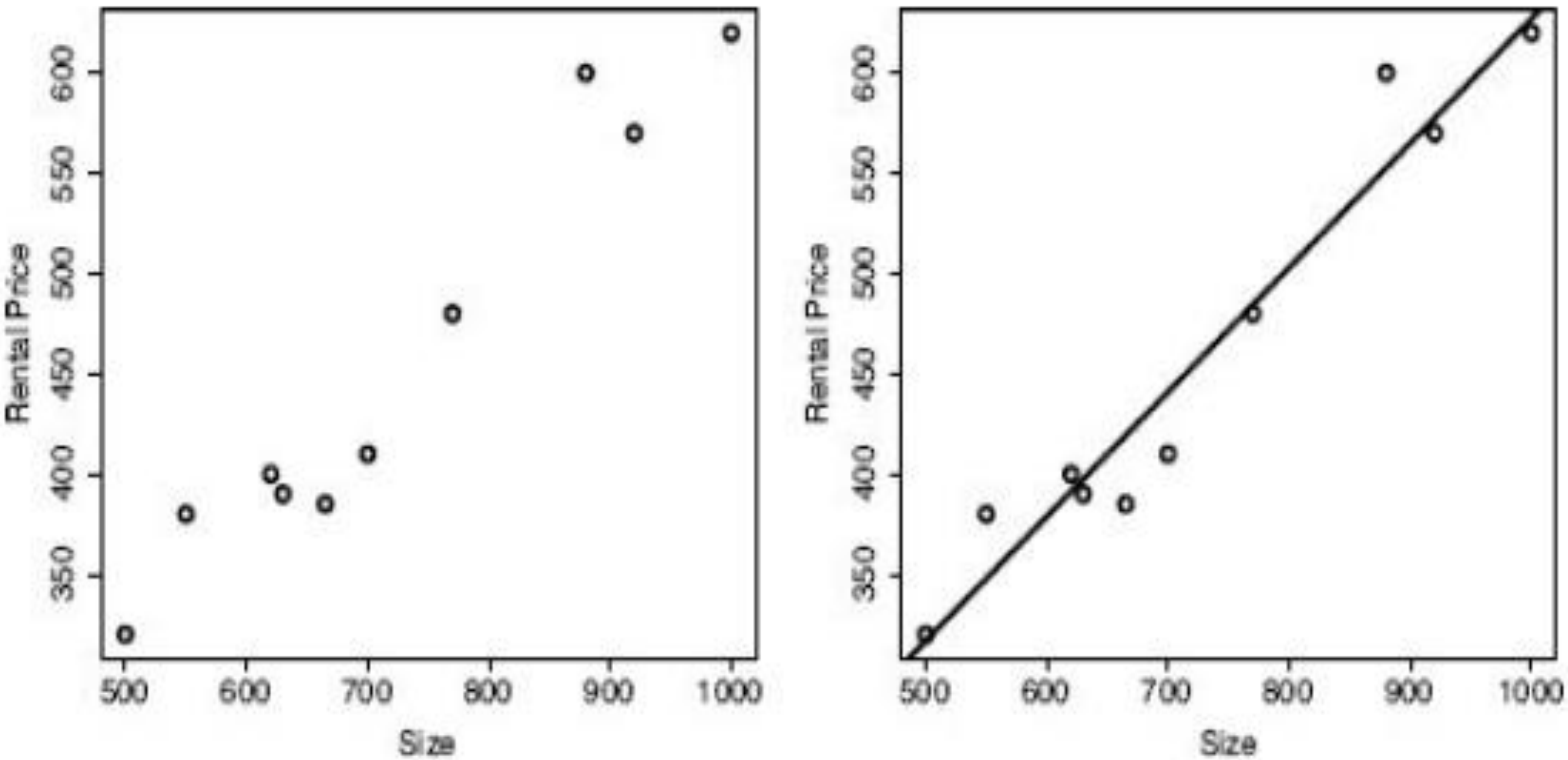
ID	Size	Floor	Broadband Rate	Energy Rating	Rental Price
1	500	4	8	C	320
2	550	7	50	A	380
3	620	9	7	A	400
4	630	5	24	B	390
5	665	8	100	C	385
6	700	4	8	B	410
7	770	10	7	B	480
8	880	12	50	A	600
9	920	14	8	C	570
10	1,000	9	24	B	620

Scatter plot of the SIZE and RENTAL PRICE features

ID	SIZE	RENTAL PRICE
1	500	320
2	550	380
3	620	400
4	630	390
5	665	385
6	700	410
7	770	480
8	880	600
9	920	570
10	1,000	620



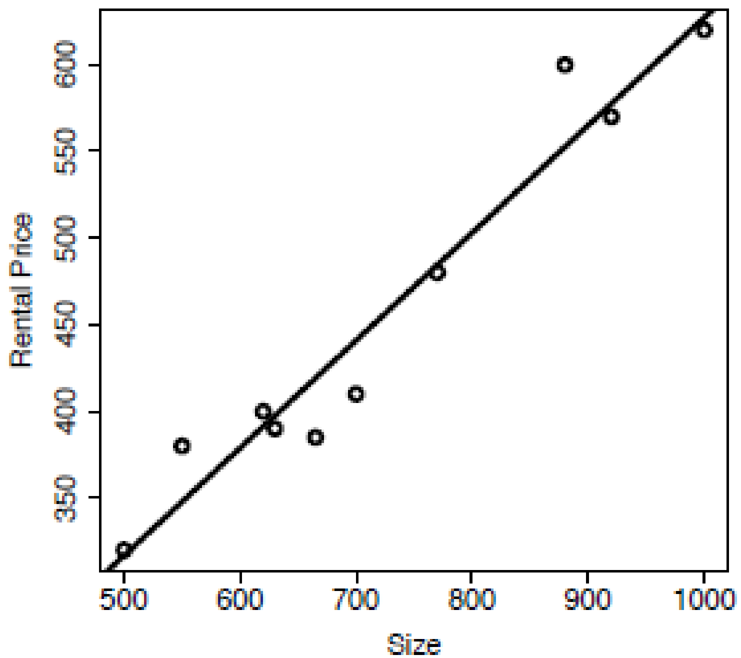
Linear Model relating Size and Rent



Linear Model

- The equation of a line can be written as: $y = mx + b$
- This model is:

$$\text{RENTAL PRICE} = 6.47 + 0.62 \;\text{SIZE}$$



Advantage

- Understand how office size affects office rental price.

- Determine the expected rental price for office sizes that we have never actually seen in the historical data

- How much would we expect for size = 730

•

Prediction

$$\textsc{Rental Price} = 6.47 + 0.62 \times \textsc{Size}$$

- Using this model determine the expected rental price of the 730 square foot office:

$$\begin{aligned}\textsc{Rental Price} &= 6.47 + 0.62 \times 730 \\ &= 459.07\end{aligned}$$

Regression Model

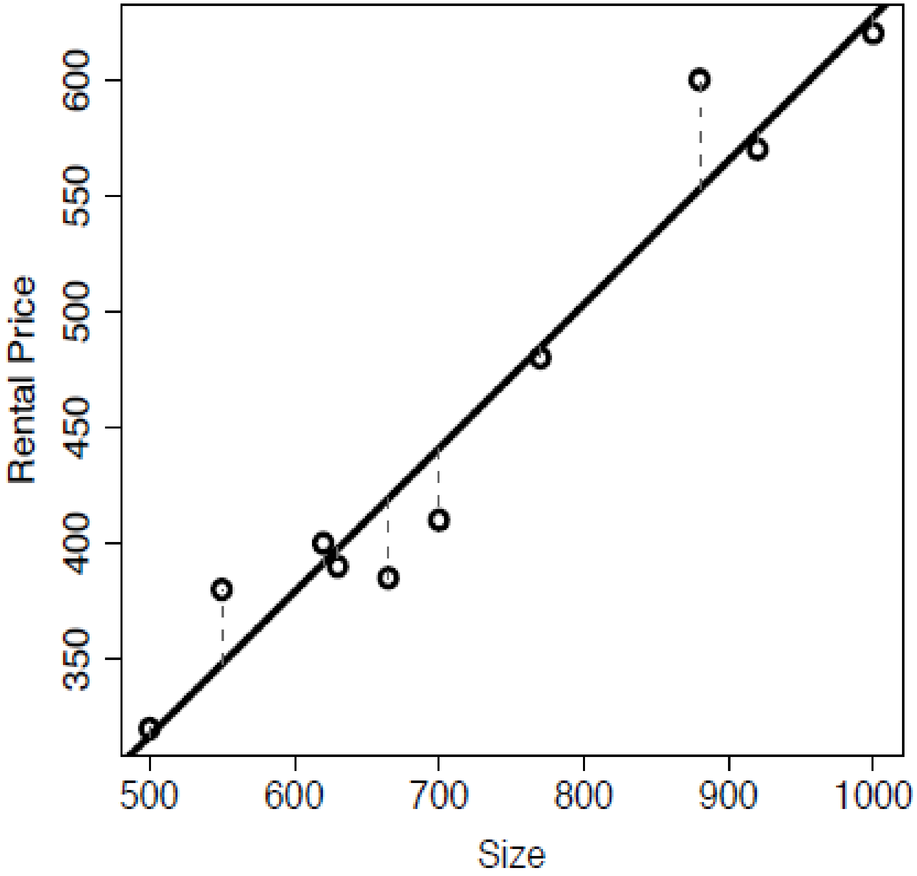
$$y = mx + b$$

$$y = w[0] + w[1] * size$$

Objective is :

- to determine the optimal values for the weights in the model
- To measure the error between the predictions a model makes and the actual rental prices

Measuring Error



$$Error\ \ E = \sum_{i=1}^{n} (t_i - y_i)^2$$

$$= \sum_{i=1}^{n} (t_i - (w[0] + w[1] * size_i))^2$$

Measuring Error

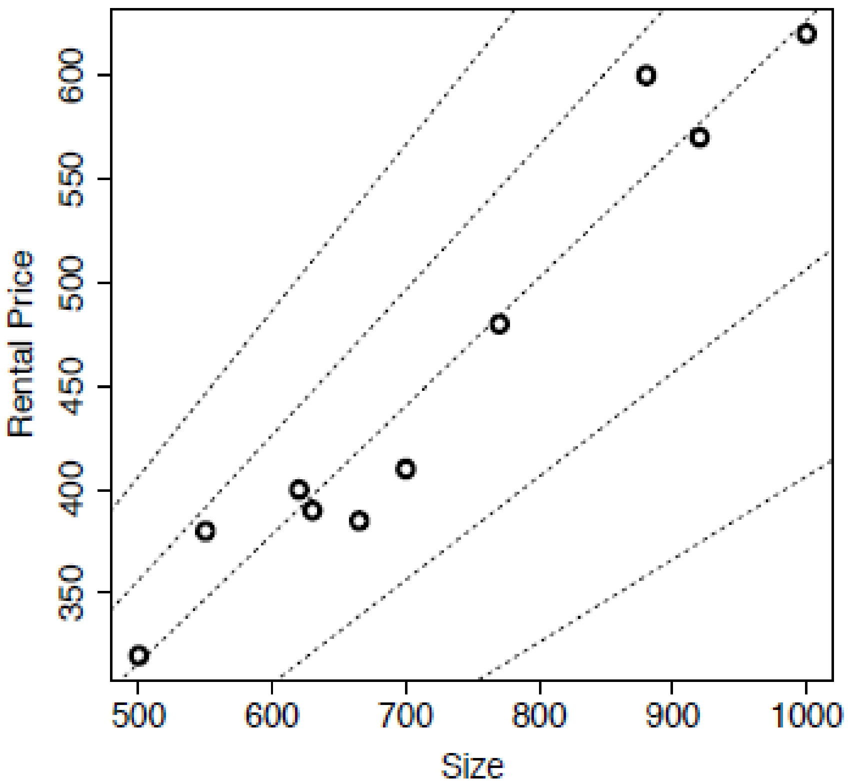
The sum of squared errors for the model (with **w[0] = 6.47 and w[1] = 0.62)**

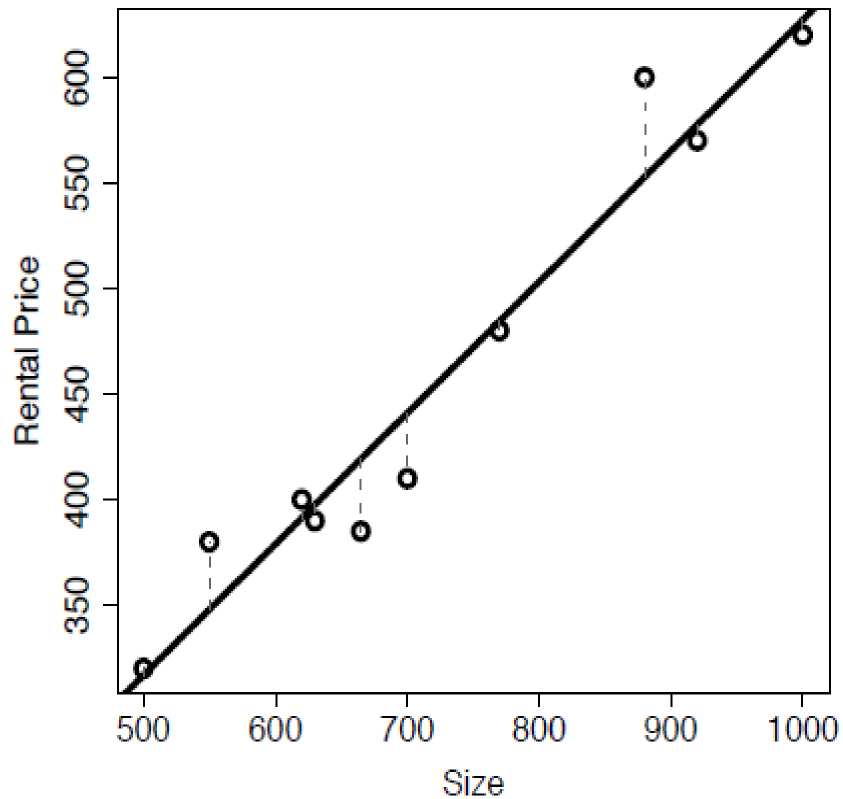
ID	Size	Rental Price
1	500	320
2	550	380
3	620	400
4	630	390
5	665	385
6	700	410
7	770	480
8	880	600
9	920	570
10	1,000	620

ID	Rental Price	Model Prediction	Error Error	Squared Error
1	320	316.79	3.21	10.32
2	380	347.82	32.18	1,035.62
3	400	391.26	8.74	76.32
4	390	397.47	-7.47	55.80
5	385	419.19	-34.19	1,169.13
6	410	440.91	-30.91	955.73
7	480	484.36	-4.36	19.01
8	600	552.63	47.37	2,243.90
9	570	577.46	-7.46	55.59
10	620	627.11	-7.11	50.51
			Sum	**5,671.64**

Collection of simple linear models

With **w[1] set to 0.4, 0.5, 0.7, and 0.8, the sums of squared errors** are 136,218, 42,712, 20,092, and 90,978



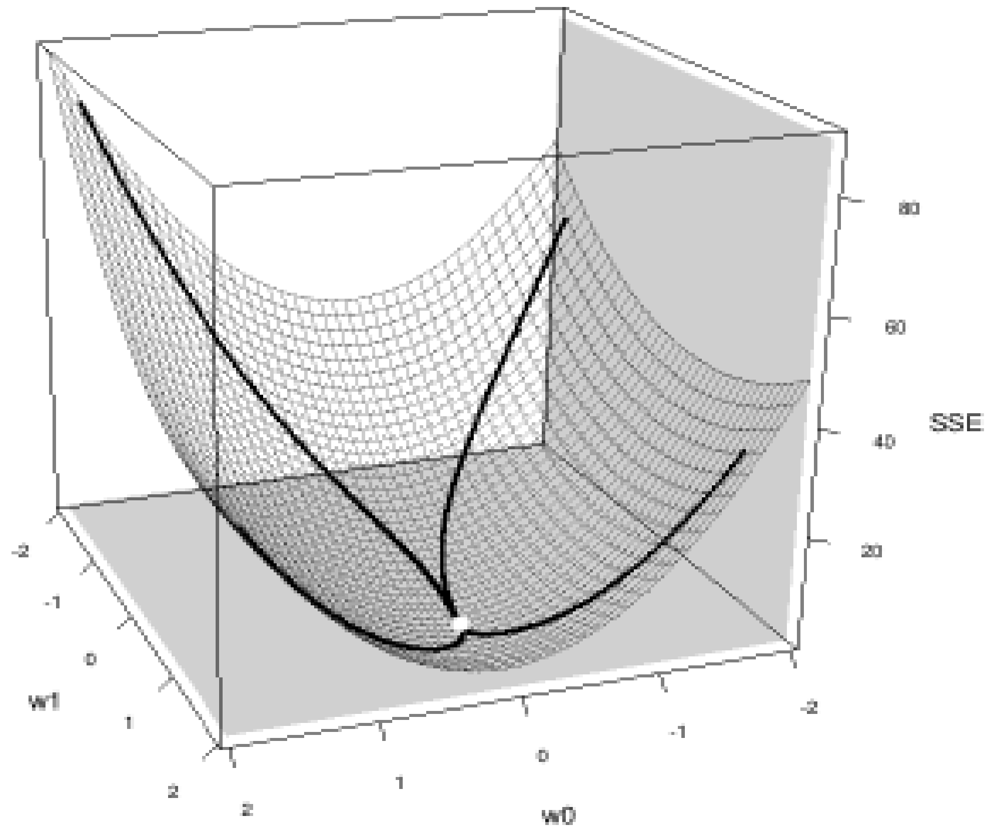


Important Observation

- The sum of squared errors function can be used to measure how well any combination of weights fits the instances in a training dataset
- From a collection of simple linear models we need to identify the one which minimizes the sum of the squared errors
- How do we identify this?

Error Surface

- For every possible combination of weights, **w[0] and w[1],** there is a corresponding sum of squared errors value that can be joined together to make a surface.



Least squares optimization.

- Having a global minimum means that on an error surface, there is a unique set of optimal weights with the lowest sum of squared errors.
- If we can find the global minimum of the error surface, we can find the set of weights defining the model that best fits the training dataset.
- This approach to finding weights is known as **least squares optimization.**

Stationary Points

- We can find the optimal weights at the point where the **partial derivatives of the error** surface with respect to **w[0] and w[1] are equal to 0**

$$E = \sum_{i=1}^{n} (t_i - (w[0] + w[1] * size_i))^2$$

$$\frac{\partial E}{\partial w[0]} = \sum_{i=1}^{n} 2(t_i - (w[0] + w[1] * size_i))(-1) \quad = 0$$

$$\sum_{i}^{n} t_i = n.w[0] + w[1] * \sum_{i}^{n} size_i$$

$$\frac{\partial E}{\partial w[1]} = \sum_{i=1}^{n} 2(t_i - (w[0] + w[1] * size_i))(size_i) \quad = 0$$

$$\sum_{i}^{n} t_i\, size_i = w[0] \sum_{i}^{n} size_i + w[1] * \left(\sum_{i}^{n} size_i \right)^2$$

Simple example

ID	Size	Rental Price
1	500	320
2	550	380
3	620	400
4	630	390
5	665	385
6	700	410
7	770	480
8	880	600
9	920	570
10	1,000	620

$$\sum_{i}^{n} t_i = n.w[0] + w[1] * \sum_{i}^{n} size_i$$

$$\sum_{i}^{n} t_i\, size_i = w[0] \sum_{i}^{n} size_i + w[1] * \left(\sum_{i}^{n} size_i \right)^2$$

$$4555 = 10w[0] + w[1] * 7235$$

$$3447725 = w[0] * 7235 + w[1] * 5479725$$

Simple example

- import numpy as np
- mat=np.array([[10, 7235], [7235, 5479725]])
- b=np.array([4555, 3447725])
- imat=np.linalg.inv(mat)
- print(imat)
- w=np.dot(imat, b)

- w=[6.46689981 0.62064008]

- This model is:

$$\text{RENTAL PRICE} = 6.47 + 0.62 \times \text{SIZE}$$

Multivariate Regression

$$y = w[0] + w[1] * size \qquad y = w[0] + w[1] * d[1]$$

$$y = w[0] + w[1] * d[1] + w[2] * d[2] + w[3] * d[3] + \cdots + w[n] * d[n]$$

Multivariate regression model equation for Rental dataset

$$\begin{aligned} \text{Rental Price} = \mathbf{w}[0] \quad &+ \quad \mathbf{w}[1] \times \text{Size} + \mathbf{w}[2] \times \text{Floor} \\ &+ \quad \mathbf{w}[3] \times \text{Broadband Rate} \end{aligned}$$

Gradient Descent

- There is, however, a simple approach to learning weights that is based on the fact that,
- The error surfaces that correspond to these high-dimensional weight spaces still have that single global minimum.
- This approach uses a guided search from a random starting position and is known as **gradient descent.**

Gradient Descent

- Gradient descent starts by selecting a random point within the weight space .

- Calculate the sum of squared errors(E) associated with this point based on predictions made for each instance in the training set .

- Determine the slope of the error surface $\dfrac{\partial E}{\partial w[j]}$

- Calculate the value of this derivative at the random point selected in the weight space.

Gradient Descent

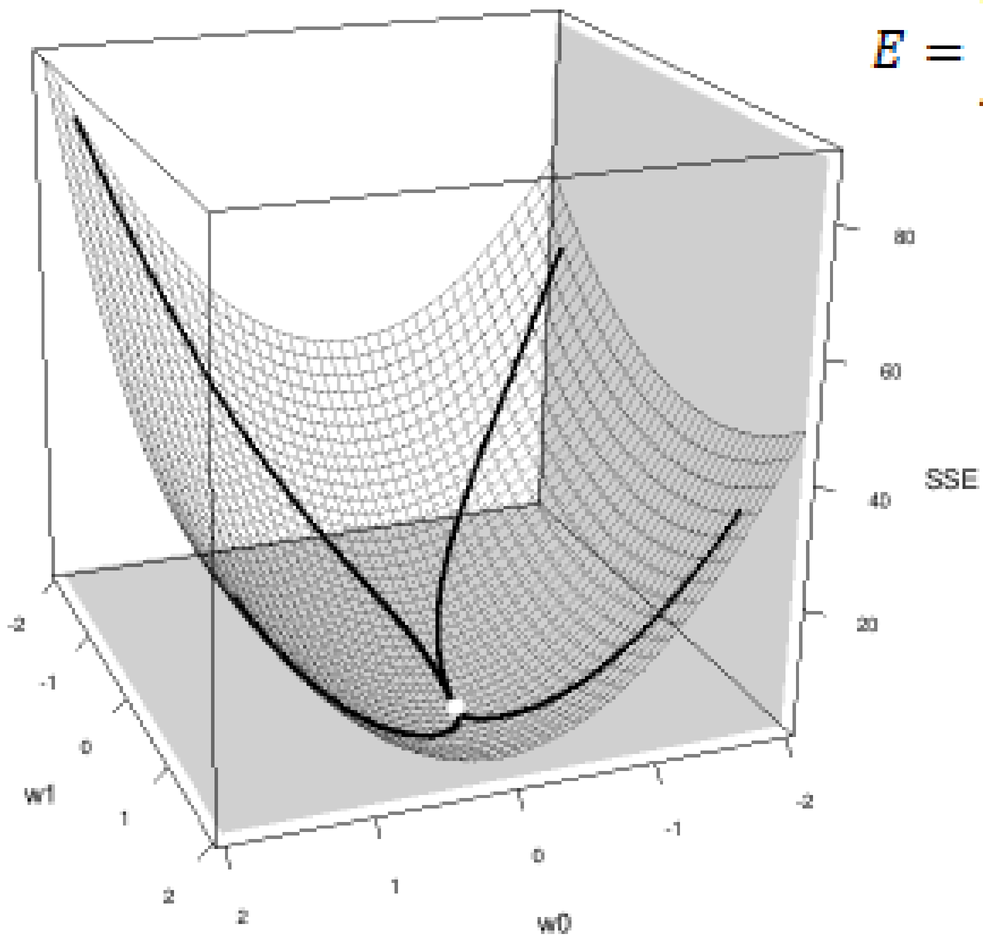
- The randomly selected weights are adjusted slightly in the direction of the error surface gradient to move to a new position on the error surface.

- Because the adjustments are made in the direction of the error surface gradient, this new point will be closer to the overall global minimum.

- This adjustment is repeated over and over until the global minimum on the error surface is reached

Gradient Descent

$$E = \sum_{i=1}^{n} (t_i - (w[0] + w[1] * size_i))^2$$



The journey across the error surface that is taken by the gradient descent algorithm when training

the simple version of the office rentals example - involving just SIZE and RENTAL PRICE.

Calculation of best fit weights for the same model by GD

- *1: w - random starting point in the weight space*
- *2: repeat*
- *3: for each w[j] in w do*

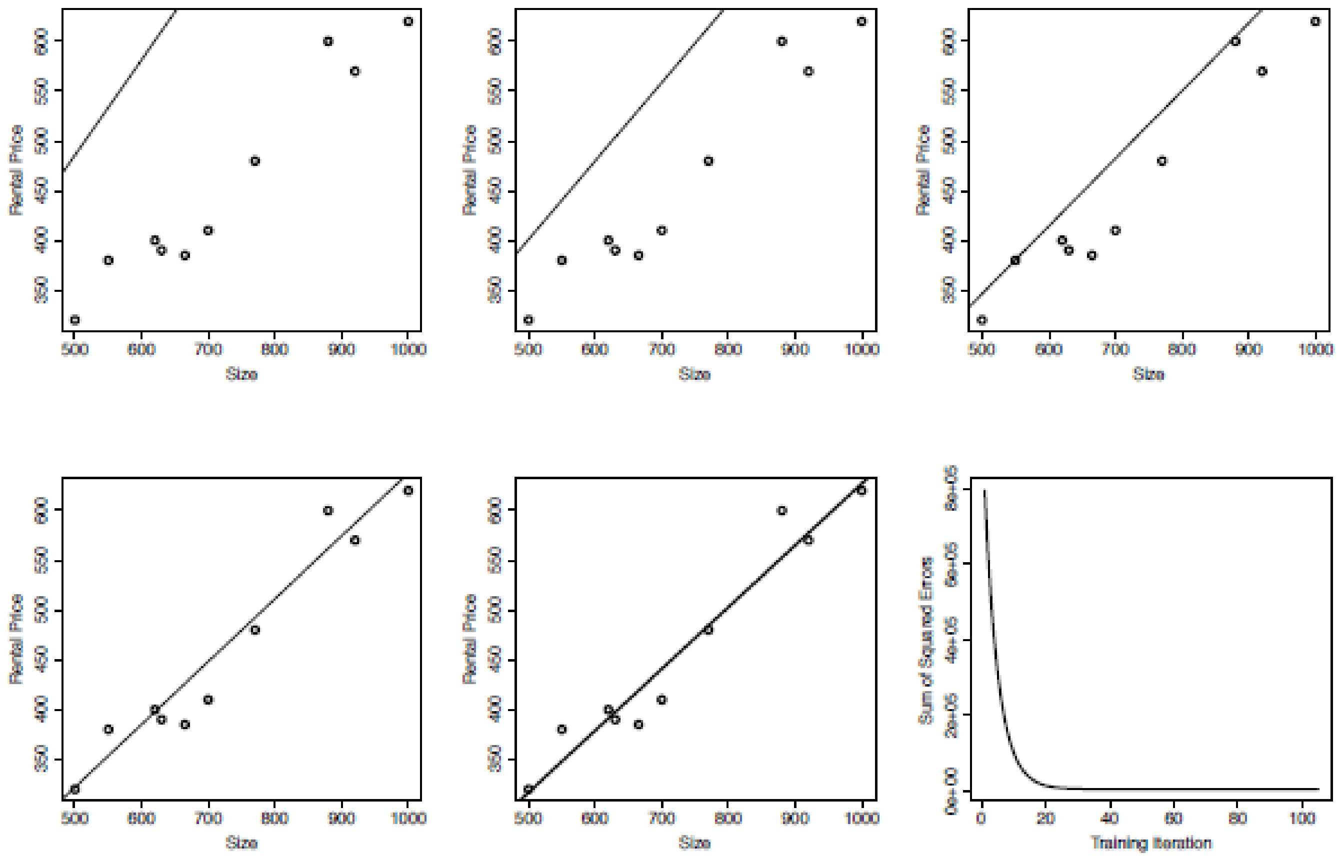
$$w[j] \leftarrow w[j] + \alpha \frac{\partial E}{\partial w[j]}$$

$$w[j] \leftarrow w[j] + \alpha \sum_{i=1}^{n} (t_i - y_i) \frac{-\partial y_i}{\partial w[j]} = \alpha \sum_{i=1}^{n} (t_i - y_i).(-d[j])$$

$$w[j] \leftarrow w[j] + \alpha * errorDelta(d[j], error)$$

- 4: end for
- 5: until convergence occurs

selection of the simple linear regression models



Multivariate Regression

$$y = w[0] + w[1] * size \qquad y = w[0] + w[1] * d[1]$$

$$y = w[0] + w[1] * d[1] + w[2] * d[2] + w[3] * d[3] + \cdots + w[n] * d[n]$$

Multivariate regression model equation for Rental dataset

$$\begin{aligned} \text{Rental Price} = \mathbf{w}[0] \; & + \; \mathbf{w}[1] \times \text{Size} + \mathbf{w}[2] \times \text{Floor} \\ & + \; \mathbf{w}[3] \times \text{Broadband Rate} \end{aligned}$$

A worked Example

ID	Size	Floor	Broadband Rate	Energy Rating	Rental Price
1	500	4	8	C	320
2	550	7	50	A	380
3	620	9	7	A	400
4	630	5	24	B	390
5	665	8	100	C	385
6	700	4	8	B	410
7	770	10	7	B	480
8	880	12	50	A	600
9	920	14	8	C	570
10	1,000	9	24	B	620

A worked Example

- For this example let's assume that:
  - $\alpha = 0.00000002$

Initial Weights

w[0]:	-0.146	**w**[1]:	0.185	**w**[2]:	-0.044	**w**[3]:	0.119

A worked Example

Iteration 1

ID	Rental Price	Pred.	Error	Squared Error	errorDelta($\mathcal{D}$, w[i]) w[0]	w[1]	w[2]	w[3]
1	320	93.26	226.74	51411.08	226.74	113370.05	906.96	1813.92
2	380	107.41	272.59	74307.70	272.59	149926.92	1908.16	13629.72
3	400	115.15	284.85	81138.96	284.85	176606.39	2563.64	1993.94
4	390	119.21	270.79	73327.67	270.79	170598.22	1353.95	6498.98
5	385	134.64	250.36	62682.22	250.36	166492.17	2002.91	25036.42
6	410	130.31	279.69	78226.32	279.69	195782.78	1118.76	2237.52
7	480	142.89	337.11	113639.88	337.11	259570.96	3371.05	2359.74
8	600	168.32	431.68	186348.45	431.68	379879.24	5180.17	21584.05
9	570	170.63	399.37	159499.37	399.37	367423.83	5591.23	3194.99
10	620	187.58	432.42	186989.95	432.42	432423.35	3891.81	10378.16
			Sum	1067571.59	3185.61	2412073.90	27888.65	88727.43

A worked Example

$$w[j] \leftarrow w[j] + \alpha errorDelta(d[j], w[j])$$

Initial Weights

w[0]:	-0.146	**w**[1]:	0.185	**w**[2]:	-0.044	**w**[3]:	0.119

Example

$$\mathbf{w}[1] \leftarrow 0.185 + 0.00000002 \times 2,412,074 = 0.23324148$$

New Weights (Iteration 1)

w[0]:	-0.146	**w**[1]:	0.233	**w**[2]:	-0.043	**w**[3]:	0.121

A worked Example

Iteration 2

ID	Rental Price	Pred.	Error	Squared Error	errorDelta($\mathcal{D}$, w[i]) w[0]	w[1]	w[2]	w[3]
1	320	117.40	202.60	41047.92	202.60	101301.44	810.41	1620.82
2	380	134.03	245.97	60500.69	245.97	135282.89	1721.78	12298.44
3	400	145.08	254.92	64985.12	254.92	158051.51	2294.30	1784.45
4	390	149.65	240.35	57769.68	240.35	151422.55	1201.77	5768.48
5	385	166.90	218.10	47568.31	218.10	145037.57	1744.81	21810.16
6	410	164.10	245.90	60468.86	245.90	172132.91	983.62	1967.23
7	480	180.06	299.94	89964.69	299.94	230954.68	2999.41	2099.59
8	600	210.87	389.13	151424.47	389.13	342437.01	4669.60	19456.65
9	570	215.03	354.97	126003.34	354.97	326571.94	4969.57	2839.76
10	620	187.58	432.42	186989.95	432.42	432423.35	3891.81	10378.16
			Sum	886723.04	2884.32	2195615.84	25287.08	80023.74
Sum of squared errors (Sum/2)				443361.52				

A worked Example

$$w[j] \leftarrow w[j] + \alpha errorDelta(d[j], w[j])$$

Initial Weights (Iteration 2)

w[0]:	-0.146	**w**[1]:	0.233	**w**[2]:	-0.043	**w**[3]:	0.121

Exercise

$\mathbf{w}[1] \leftarrow ?, \alpha = 0.00000002$

New Weights (Iteration 2)

w[0]:	?	**w**[1]:	?	**w**[2]:	?	**w**[3]:	?

A worked Example

$$w[j] \leftarrow w[j] + \alpha errorDelta(d[j], w[j])$$

Initial Weights (Iteration 2)

w[0]:	-0.146	**w**[1]:	0.233	**w**[2]:	-0.043	**w**[3]:	0.121

Exercise

w[1] $\leftarrow ?, \alpha = 0.00000002$

New Weights (Iteration 2)

w[0]:	?	**w**[1]:	?	**w**[2]:	?	**w**[3]:	?

New Weights (Iteration 2)

w[0]:	-0.145	**w**[1]:	0.277	**w**[2]:	-0.043	**w**[3]:	0.123

A worked Example

- After 100 iterations the final values for the weights are:
  - $\mathbf{w}[0] = -0.1513$,
  - $\mathbf{w}[1] = 0.6270$,
  - $\mathbf{w}[2] = -0.1781$
  - $\mathbf{w}[3] = 0.0714$

Prediction

- Using this model:

$$\begin{aligned} \textsc{Rental Price} = -0.1513 &+ 0.6270 \times \textsc{Size} \\ &- 0.1781 \times \textsc{Floor} \\ &+ 0.0714 \times \textsc{Broadband Rate} \end{aligned}$$

- we can, for example, predict the expected rental price of a 690 square foot office on the 11^{th} floor of a building with a broadband rate of 50 Mb per second as:

$$\begin{aligned} \textsc{Rental Price} &= -0.1513 + 0.6270 \times 690 \\ &\quad -0.1781 \times 11 + 0.0714 \times 50 \\ &= 434.0896 \end{aligned}$$

Linear Regression with Scikit

- Load Dataset and split target and Features
- Data Preprocessing(null values, scaling)
- Exploratory Data Analysis
- Split data into training and test part
- Fit the Model to the training data
- Print model coefficients
- Make predictions on the test data
- Compute the mean squared error
- Make Predictions for new query