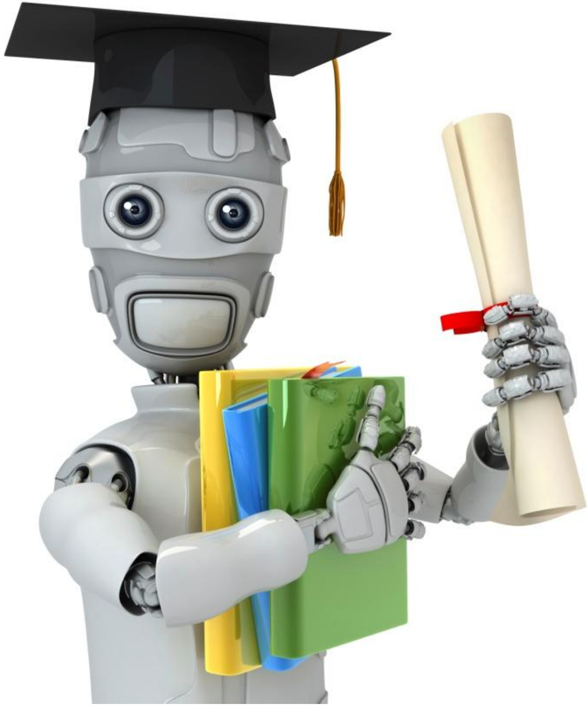


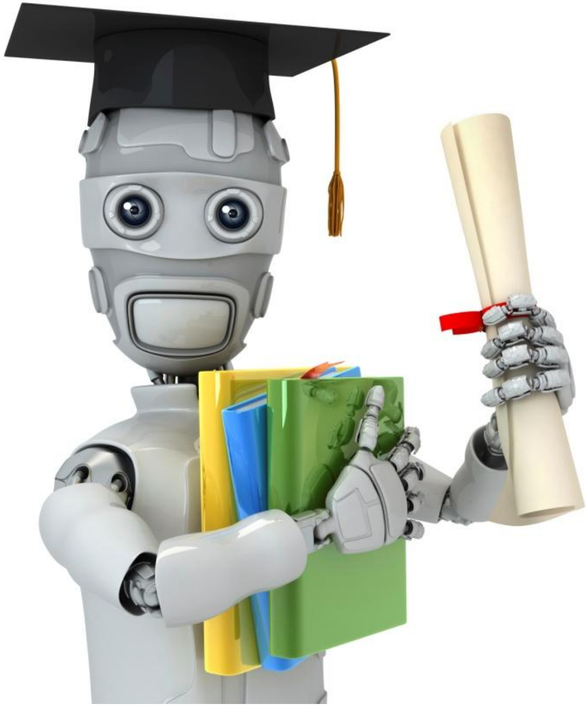
Machine Learning

Linear regression

Dr. Muhammad Sajjad
Associate Professor
Department of Computer Science
Islamia College Peshawar, Pakistan

Overview

- Recap of the Last Lecture (Supervise and Unsupervised Machine Learning Approaches)
- What is Linear Regression
- Why use Linear Regression
  - Why do we care about
  - Why is this model useful
- How does Linear Regression Work
  - Linear Regression with one variable (Model Representation)
  - Cost Function
  - Optimization Technique (Gradient Descent)



Machine Learning

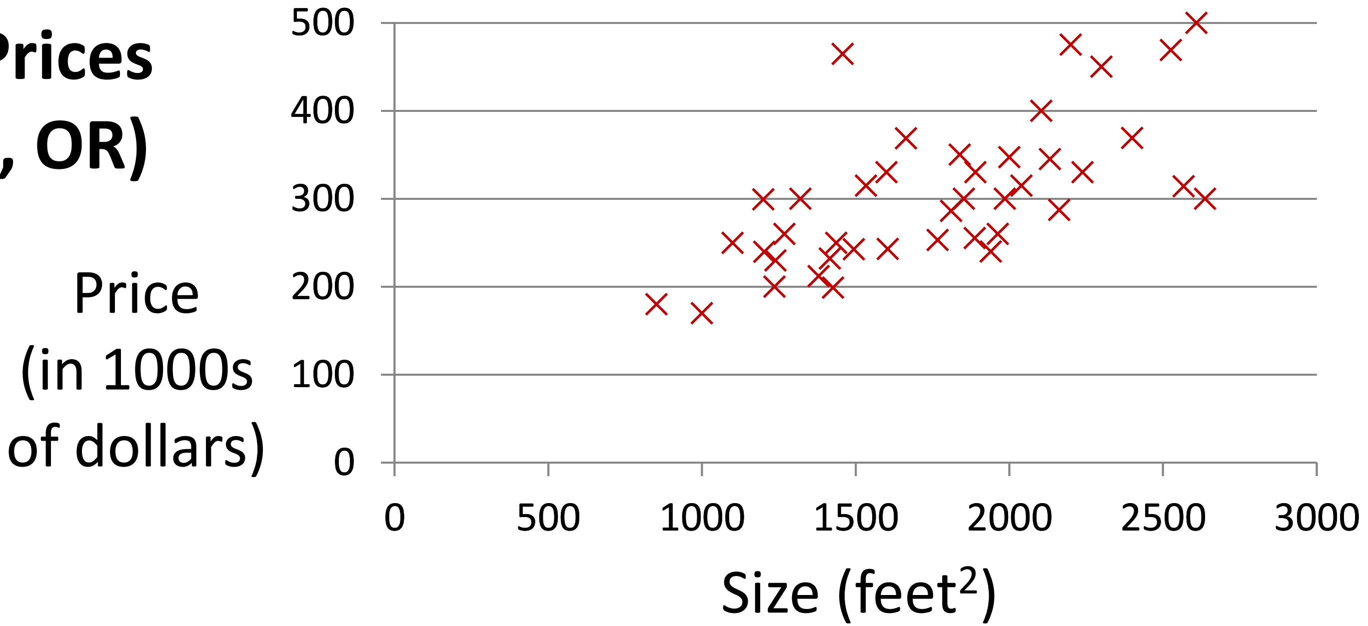
Linear regression

What?

What is Linear Regression

- Linear regression is a supervised learning algorithm used to predict a continuous value.
- It's a method to model the relationship between a dependent variable (what we're trying to predict) and one or more independent variables (the inputs that help make the prediction).
- In a simple linear regression, we aim to fit a line that best represents the relationship between two variables:
    - **Independent variable (X)**: The input or feature(s).
    - **Dependent variable (Y)**: The output or label that we want to predict.

Housing Prices (Portland, OR)

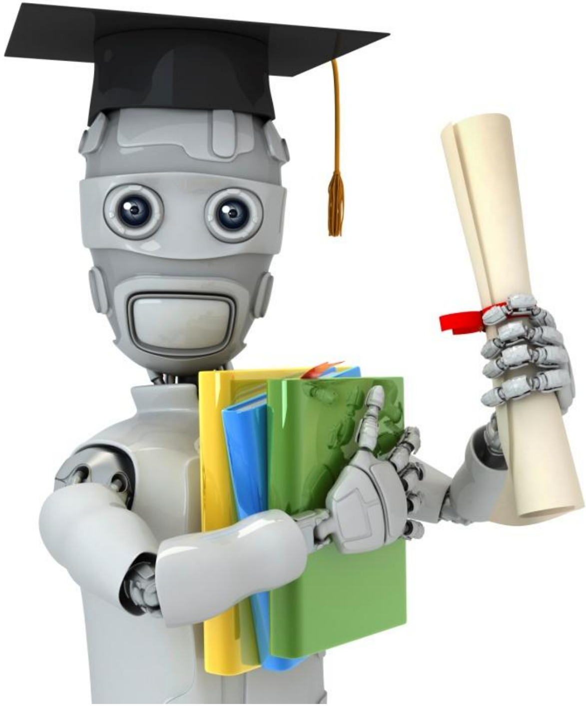


<u>Supervised Learning</u>

Given the “right answer” for each example in the data.

<u>Regression Problem</u>

Predict real-valued output



Machine Learning

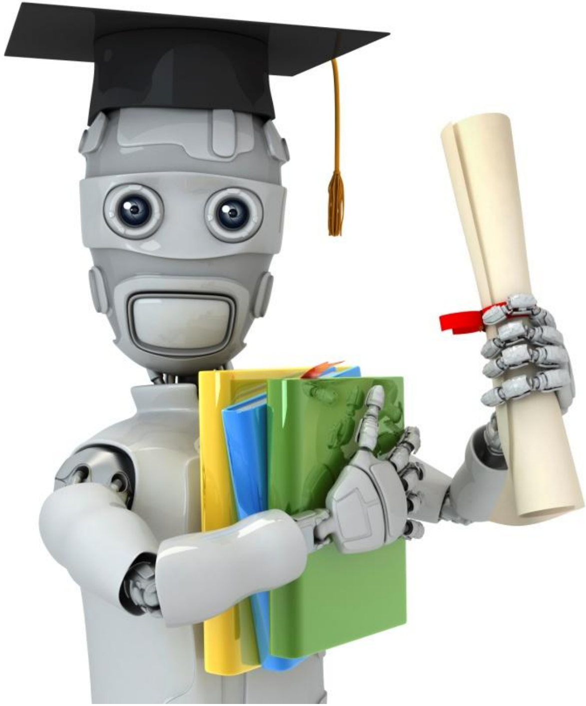
Linear regression

Why?

Why Use Linear Regression

Linear regression is one of the simplest and most interpretable machine learning models. It helps us answer critical questions like:

- **Why do we care about predicting continuous outcomes?** Continuous predictions are necessary in numerous real-world applications. For example, predicting housing prices based on square footage, stock prices based on historical data, or even sales forecasting.
- **Why is this model useful?** It provides a baseline model that is easy to understand and interpret. When you begin exploring complex datasets, linear regression often serves as a good starting point because it's fast, simple, and interpretable.
- **Applications of Linear Regression**:
  - Predicting the price of real estate based on various factors like size, location, etc.
  - Forecasting sales or stock prices.
  - Estimating healthcare costs based on patient data (age, BMI, etc.).
  - In computer science, we often use it to model relationships in data science, economic forecasting, and algorithm efficiency.



Machine Learning

Linear regression with one variable

Model representation

Training set of housing prices (Portland, OR)

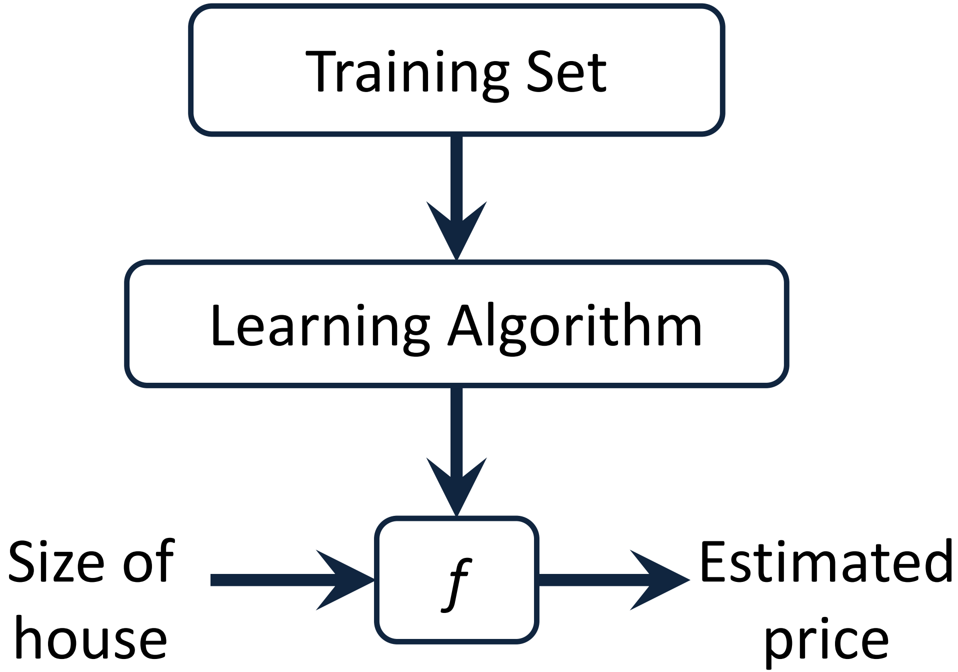
Size in feet^2 (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

Notation:

m = Number of training examples

x’s = “input” variable / features

y’s = “output” variable / “target” variable



How do we represent *f* ?

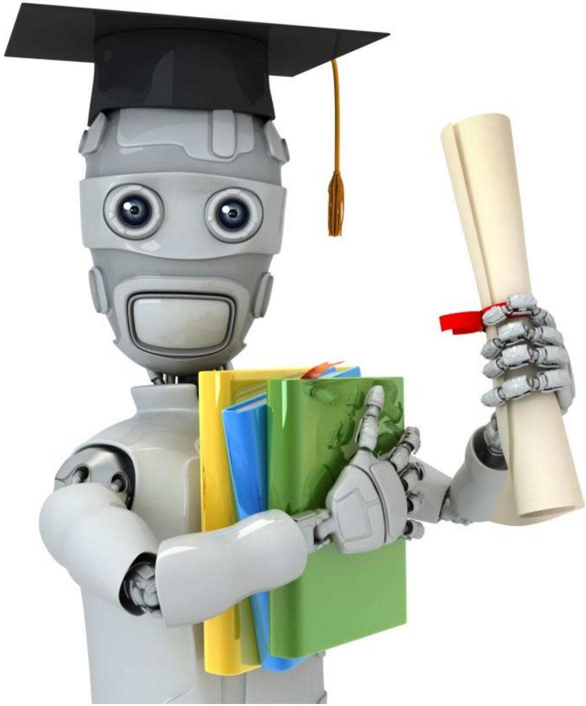
Data

Model: $f_{w,b}(x) = wx + b$

w, b: parameters

Linear regression with one variable.

Univariate linear regression.



Linear regression with one variable

Cost function

Machine Learning

Training Set

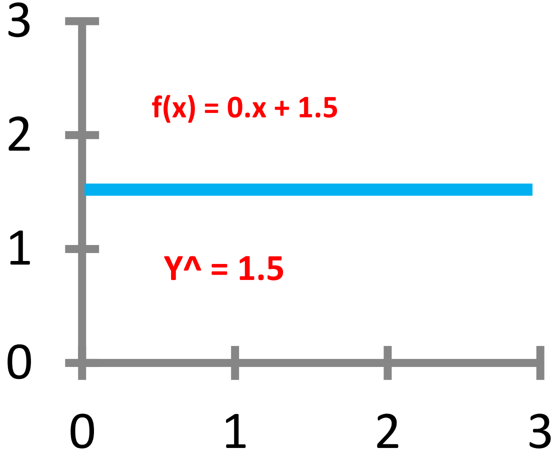
Size in feet^2 (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

Hypothesis: $f_{w,b}(x) = wx + b$

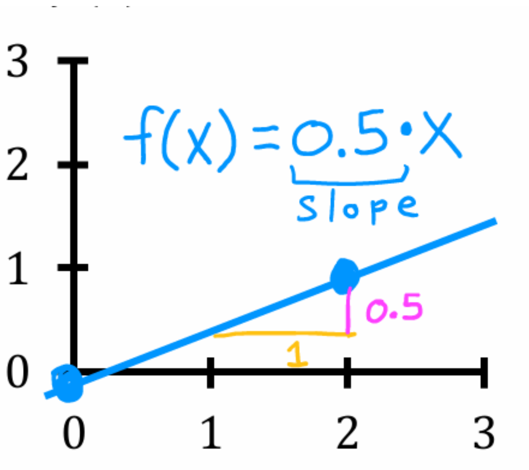
w, b: parameters

What to do w and b?

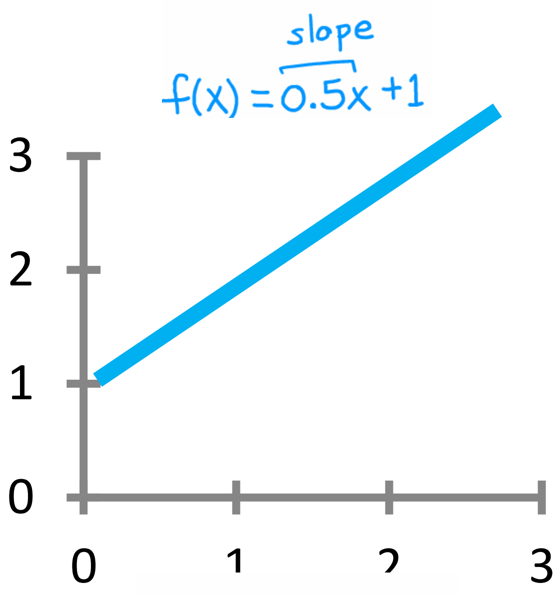
$$f_{w,b}(x) = wx + b$$



$w = 0$
$b = 1.5$



$w = 0.5$
$b = 0$



$w = 0.5$
$b = 1$

- **Understanding w (Slope of the Line)**
- The slope w determines how much y changes when x increases by one unit. It represents the rate of change of the dependent variable.

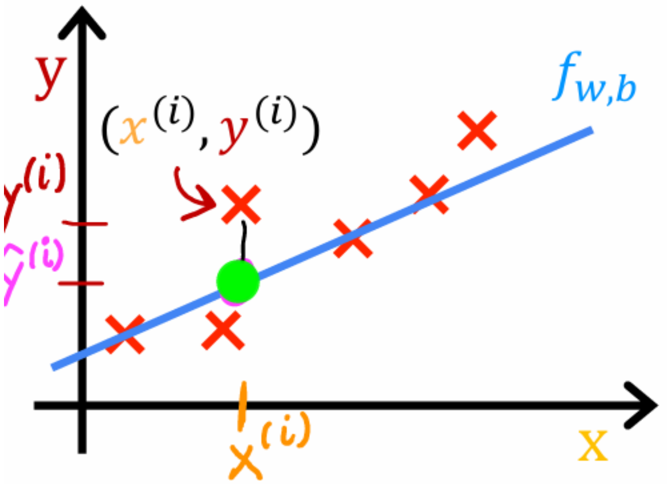
- If $w>0 \rightarrow$ y increases as x increases (positive correlation).
- If $w < 0 \rightarrow$ y decreases as x increases (negative correlation).
- If $w=0 \rightarrow$ y remains constant, meaning x has no effect on y.

- **Understanding bias:**
- The intercept **b** is the value of **y** when **x=0**. It tells us where the line crosses the y-axis.
- **Interpretation of b:**
- If b>0→ The line starts above the origin.
- If b<0→ The line starts below the origin.
- If b=0 → The line passes through the origin

- ## Cost function

- In **Linear Regression**, our goal is to find the best-fitting straight line that predicts the output y based on input x. To determine how well our model is performing, we use a **Cost Function**, which measures the difference between predicted values and actual values.
- To evaluate how well our model is performing, we need a way to measure the error (difference between predicted and actual values).
- For example, suppose we have the following dataset:
- If we randomly choose **w=0.5 and b=1**, then the **predicted** values are:
- **f(x)=(0.5.x)+1**

- **Cost function**



$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

$$\hat{y}^{(i)} = f_{w,b}(x^{(i)})$$

- **Square error cost function**

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(\underbrace{\hat{y}^{(i)} - y^{(i)}}_{\text{error}}\right)^2$$

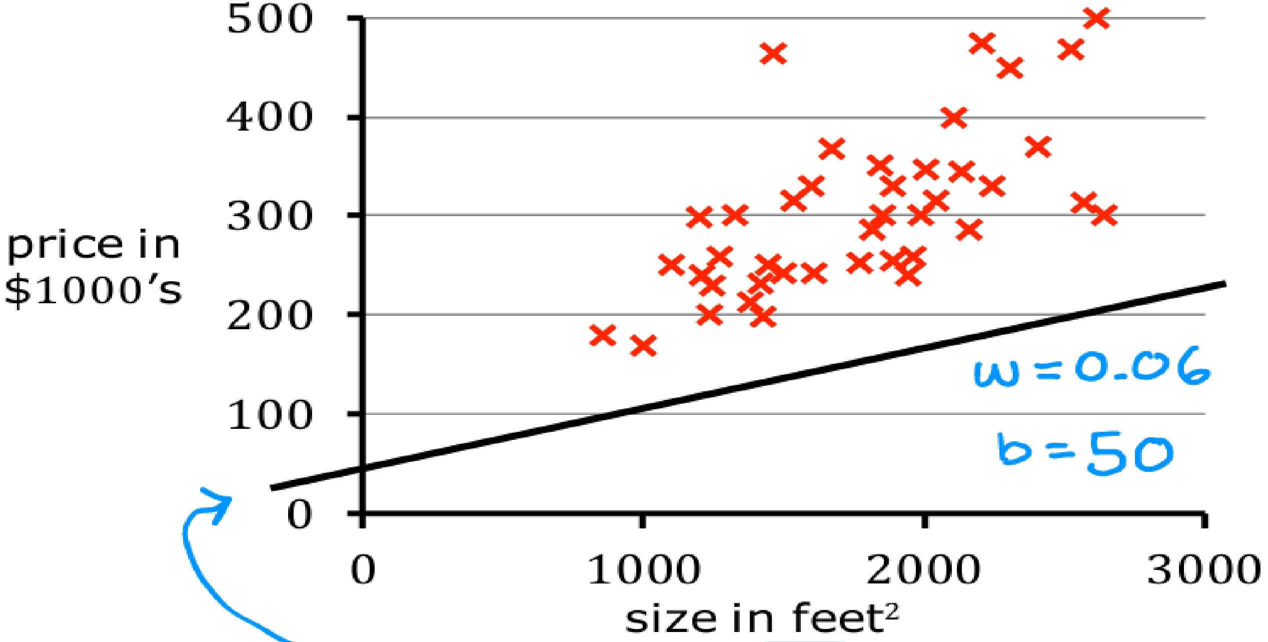
m = number of training examples

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2$$

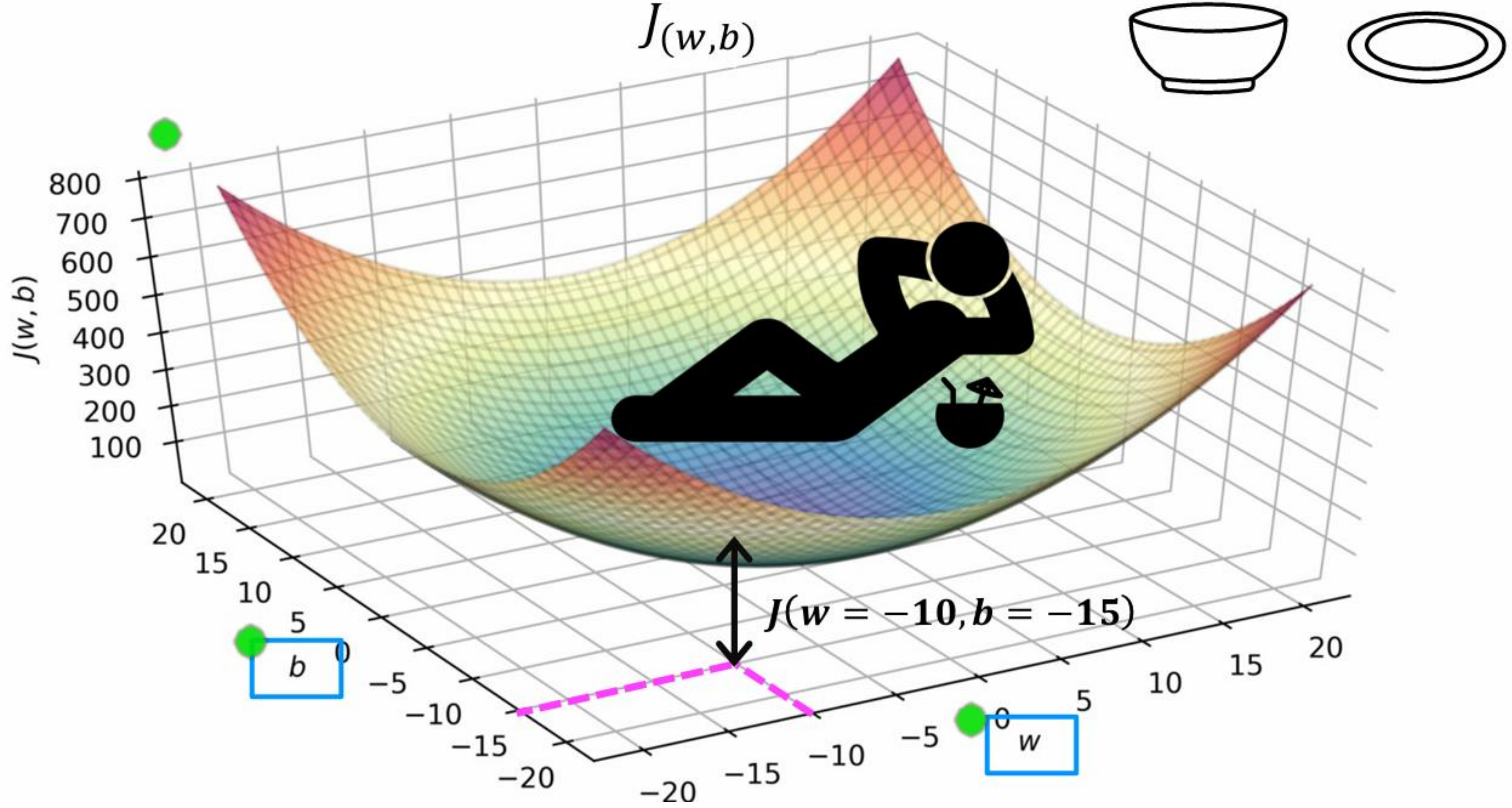
Find w, b:
$\hat{y}^{(i)}$ is close to $y^{(i)}$ for all $(x^{(i)}, y^{(i)})$.

$f_{w,b}$

(function of x)



$$\boxed{f_{w,b}(x) = 0.06x + 50}$$



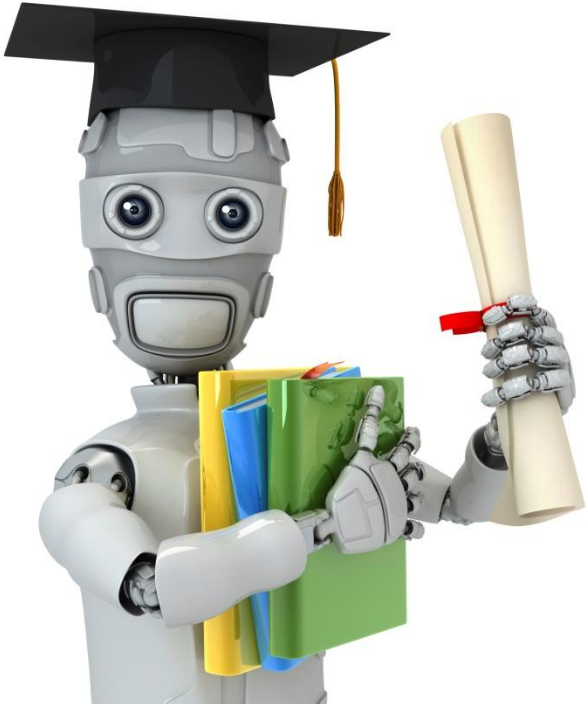
$J_{(w,b)}$
800
700
600
500
400
300
200
100
J(w, b)
20
15
10
5
0
−5
−10
−15
−20
b
w
$J(w = -10, b = -15)$

Model $f_{w,b}(x) = wx + b$

Parameters w, b

Cost Function $J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$

Objective $\underset{w,b}{\text{minimize}}\, J(w,b)$



Machine Learning

Linear regression with one variable

Gradient descent

- ## Gradient descent

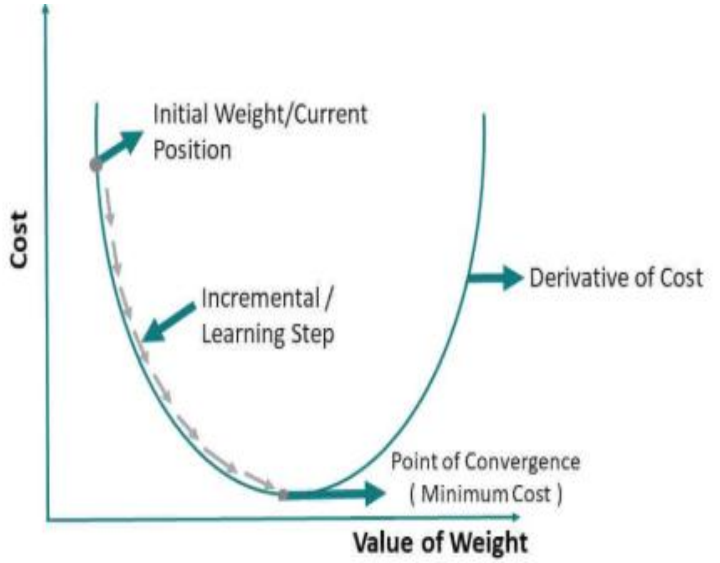
- Gradient Descent is an optimization algorithm used to minimize the cost function(error) in Machine Learning models. In Linear Regression, it helps find the optimal values of w and b that minimize the error.
- For Simple Linear Regression, the hypothesis function is:

$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

- **Cost fu**

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

- Our goal is to find w and b that $\underset{w,b}{\text{minimize}}\, J(w,b)$



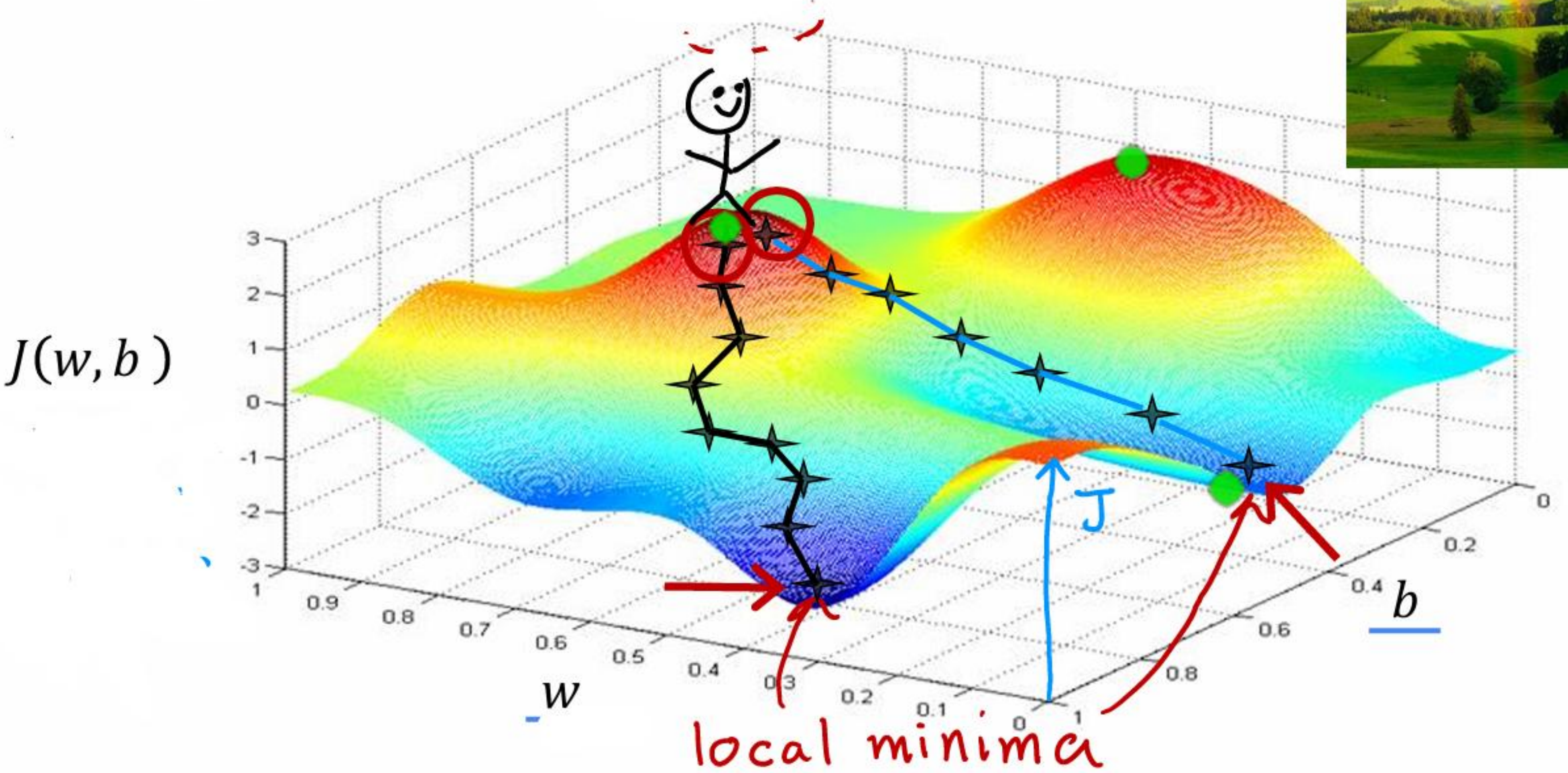
Have some function $J(w, b)$

Want $\underset{w,b}{\text{minimize}}\, J(w, b)$

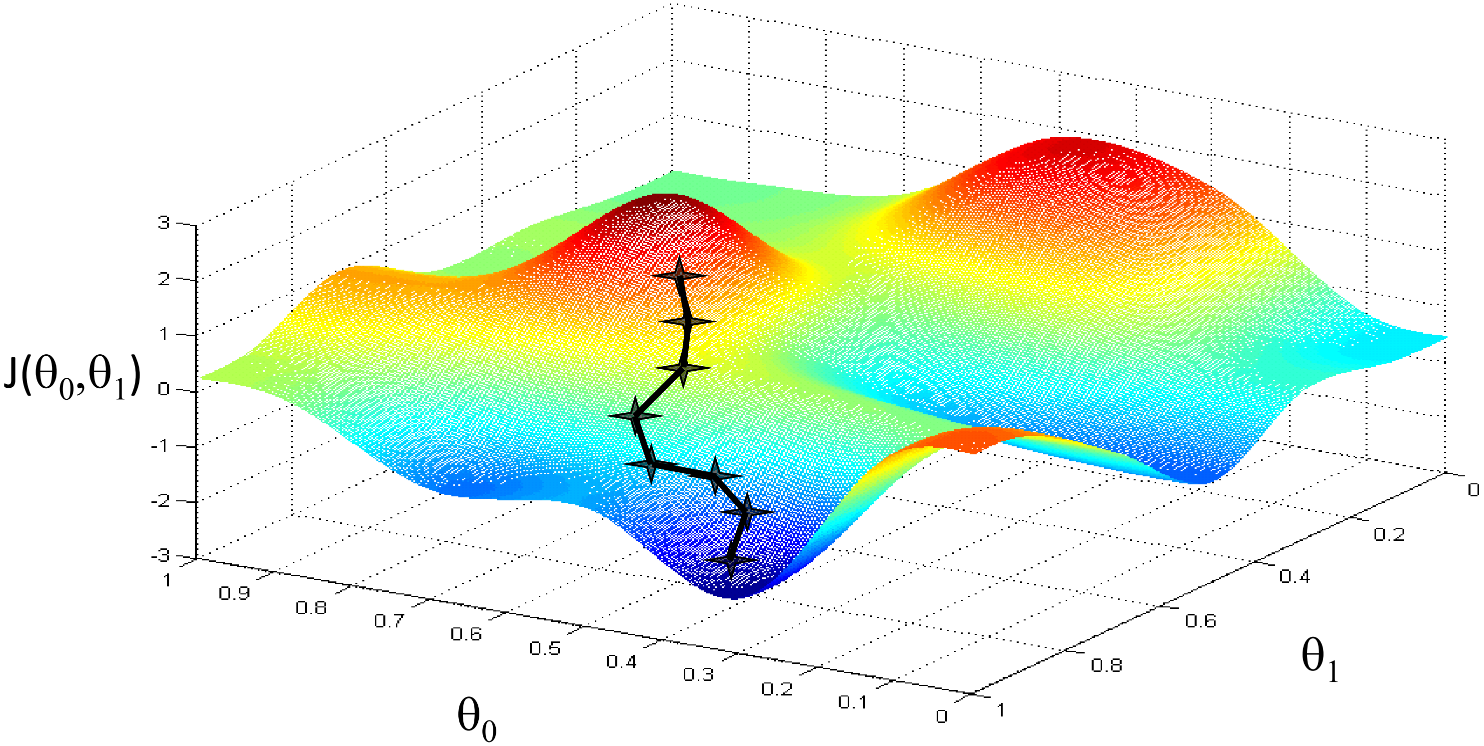
Outline:

- Start with some w , b set(w=0 and b =0)
- Keep changing w and b to reduce $J(w, b)$

  until we hopefully end up at a minimum



J(w, b)
3
2
1
0
-1
-2
-3
w
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
b
0
0.2
0.4
0.6
0.8
1
J
local minima



J(θ0,θ1)
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1

Gradient descent algorithm

Linear regression algorithm

$$f_{w,b}(x) = wx + b$$

Cost function

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})^2$$

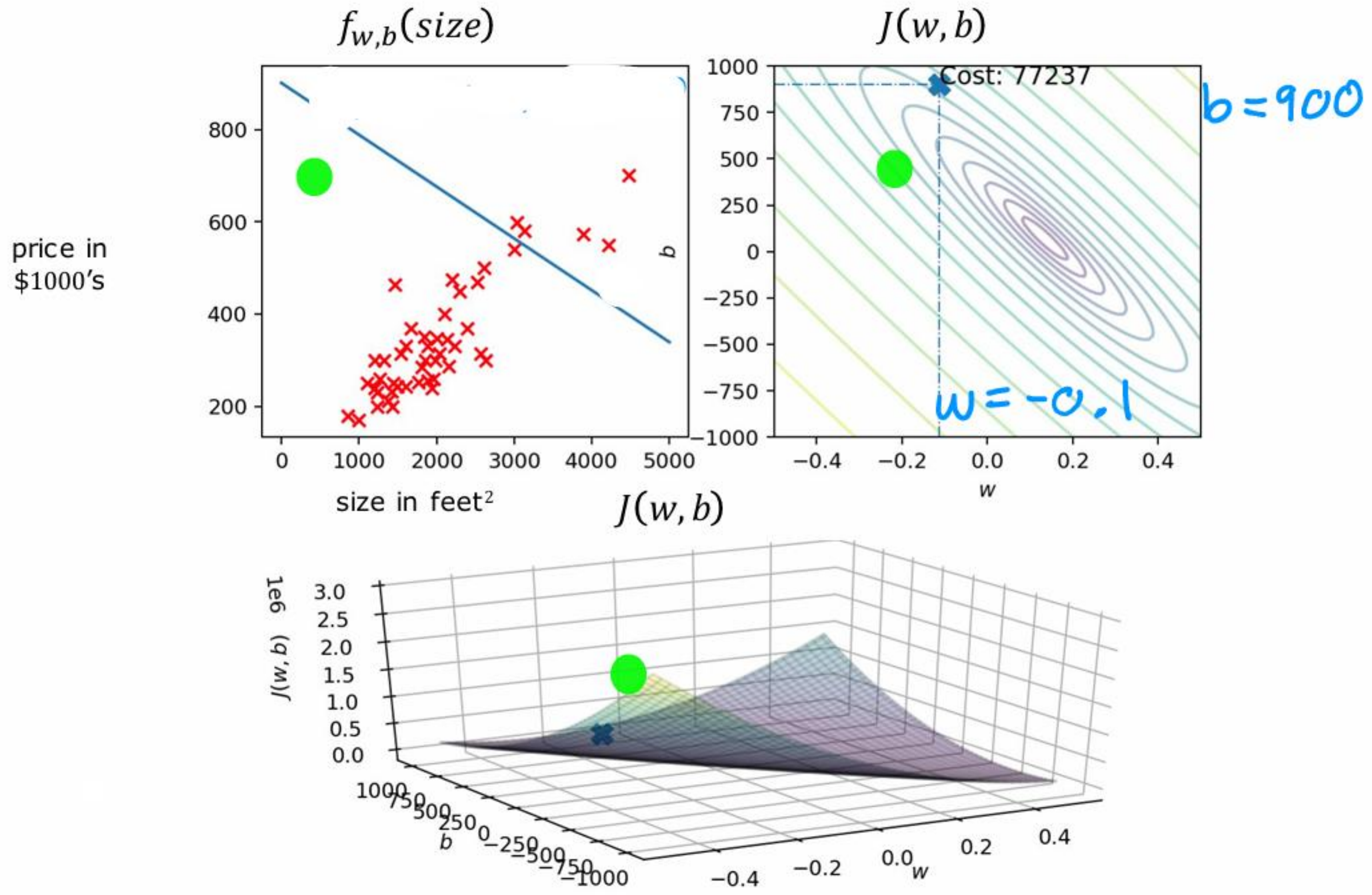
Correct: Simultaneous update

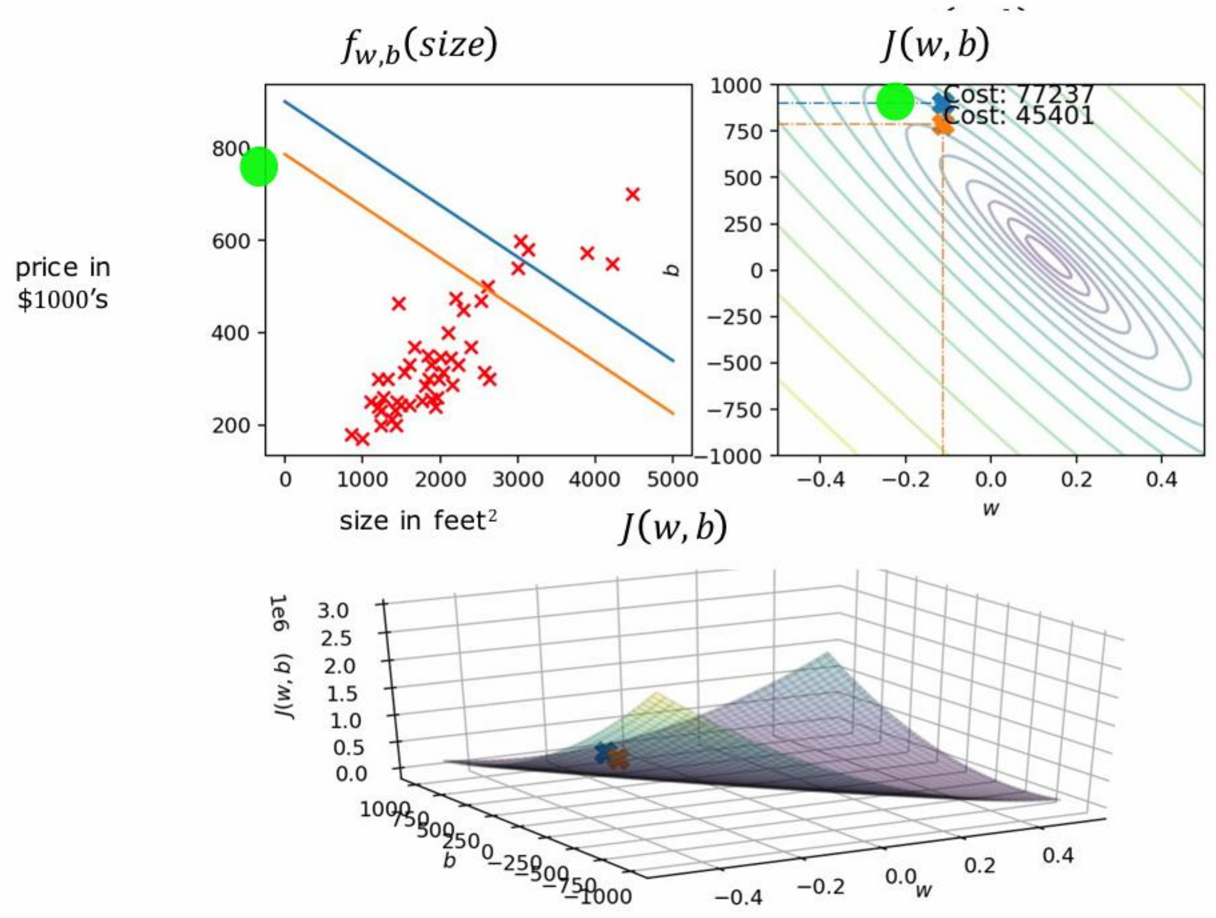
repeat until convergence {

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b) \longrightarrow \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})x^{(i)}$$

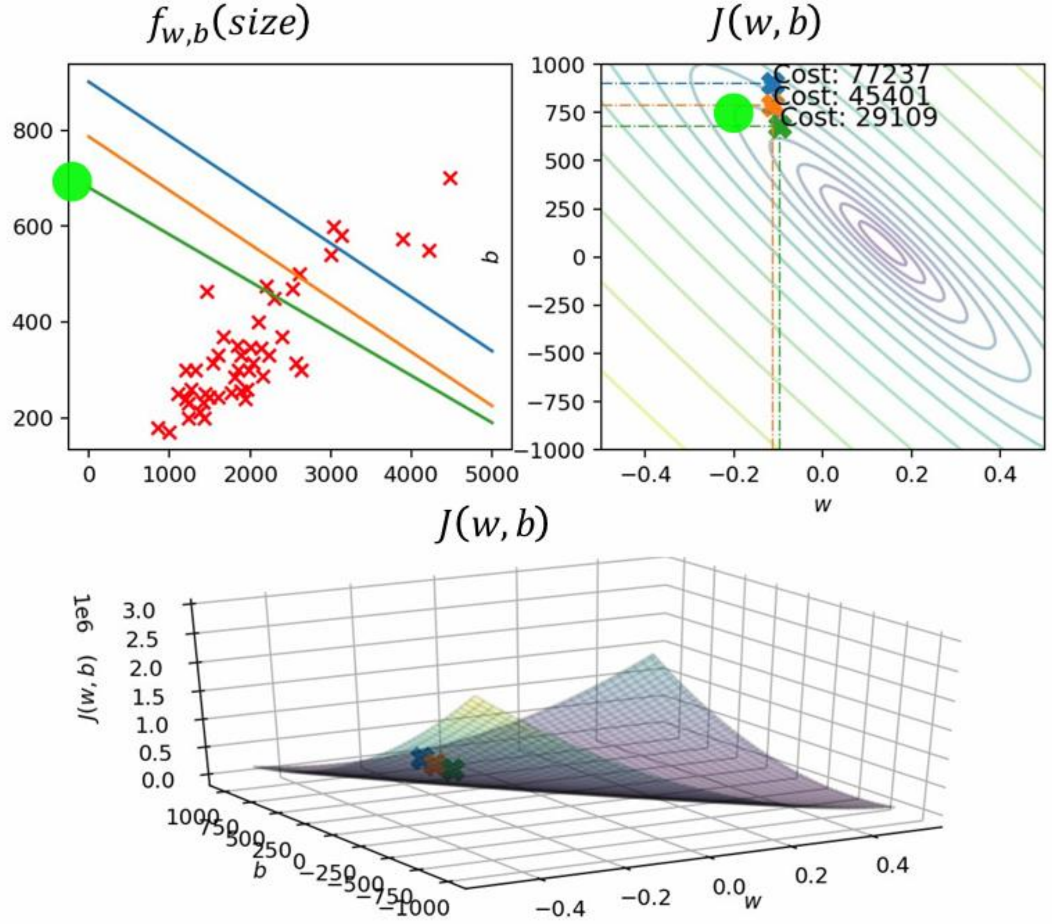
$$b = b - \alpha \frac{\partial}{\partial b} J(w,b) \longrightarrow \frac{1}{m}\sum_{i=1}^{m}(f_{w,b}(x^{(i)}) - y^{(i)})$$

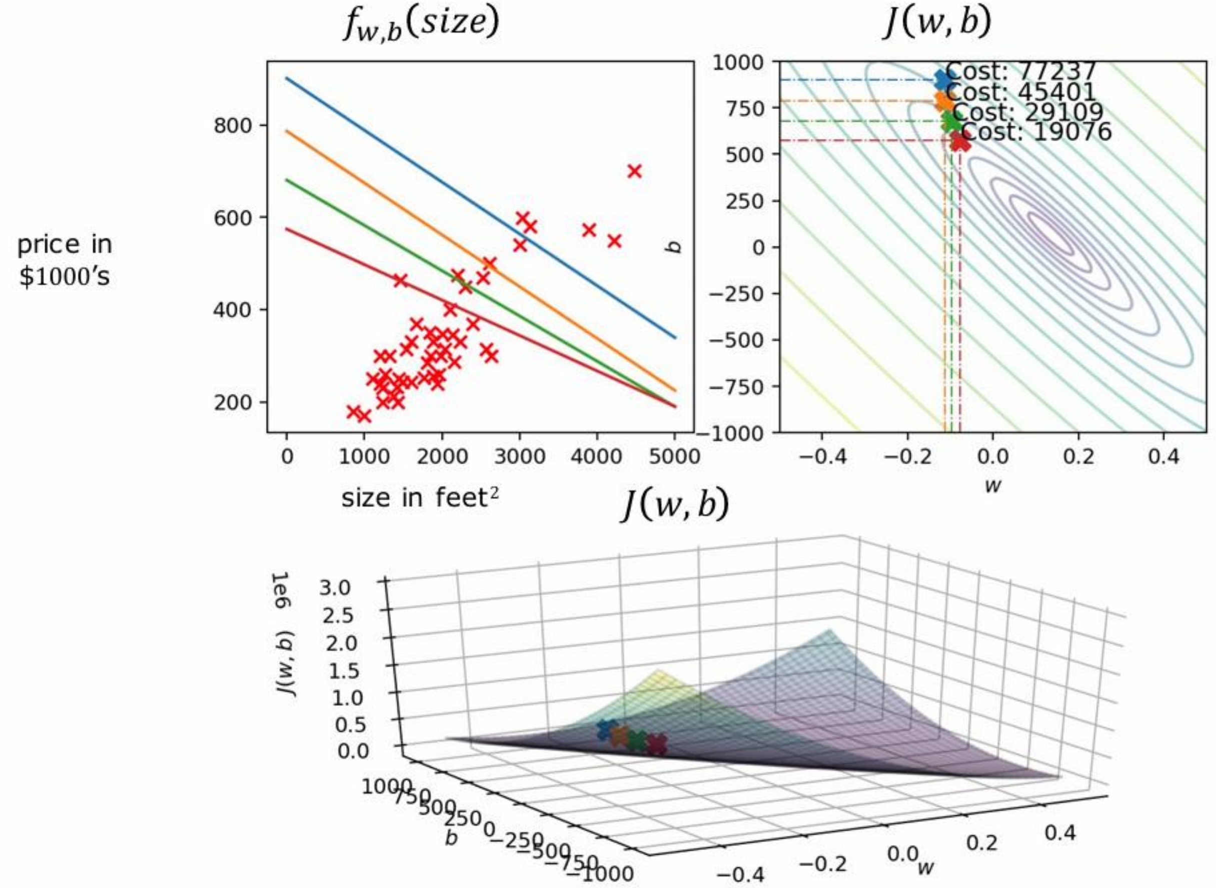
}


$f_{w,b}(size)$
$J(w,b)$
price in $1000's
size in feet^2
Cost: 77237
b=900
w=-0.1
w
b
$J(w,b)$
1e6
$J(w,b)$

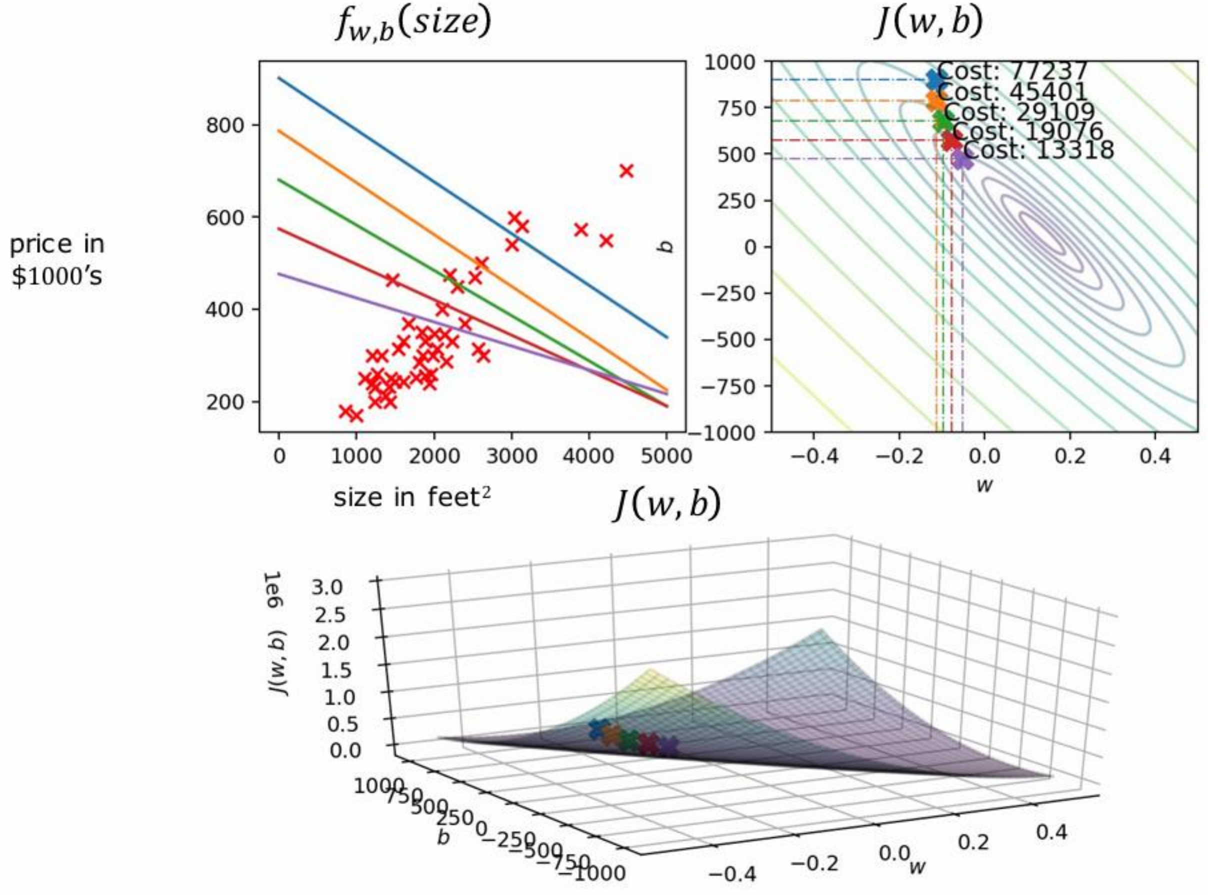


$f_{w,b}(size)$
$J(w,b)$
price in $1000's
size in feet^2
Cost: 77237
Cost: 45401
b
w
$J(w,b)$
1e6

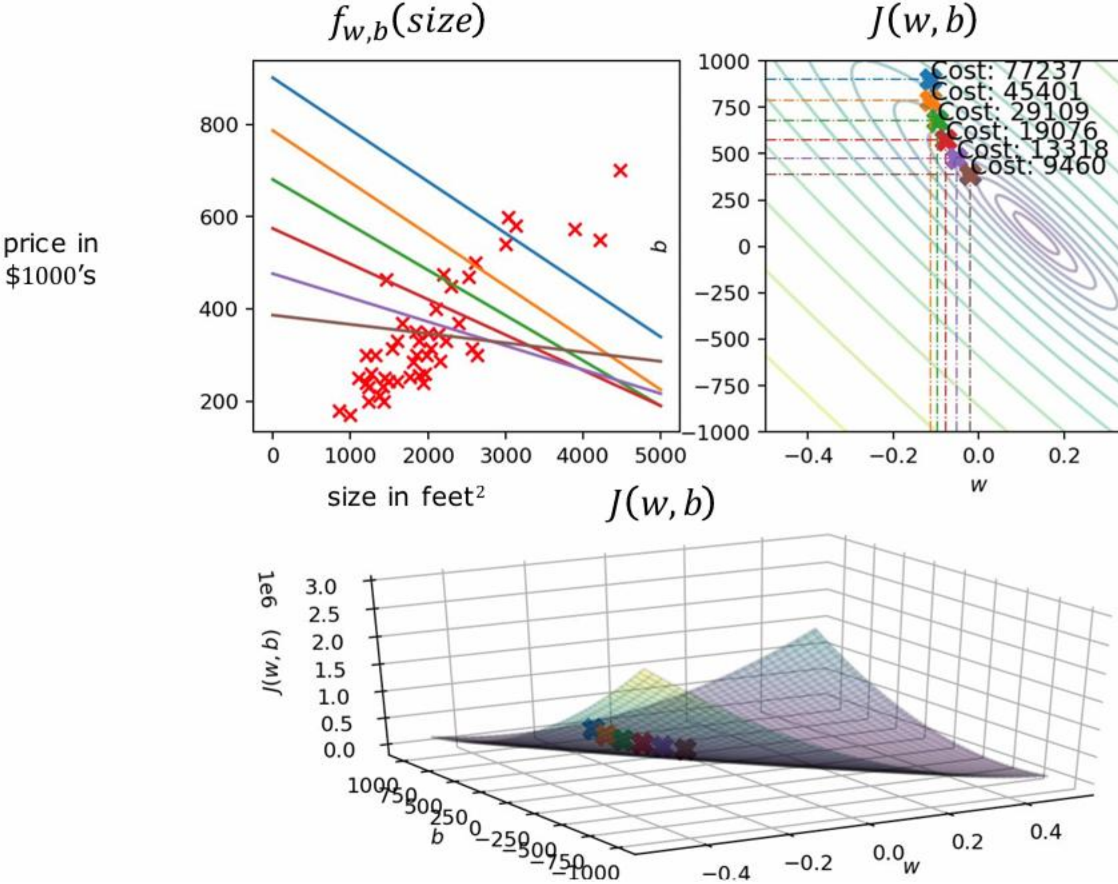

$f_{w,b}(size)$
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
b
w
$J(w,b)$
1e6
J(w, b)
b
w



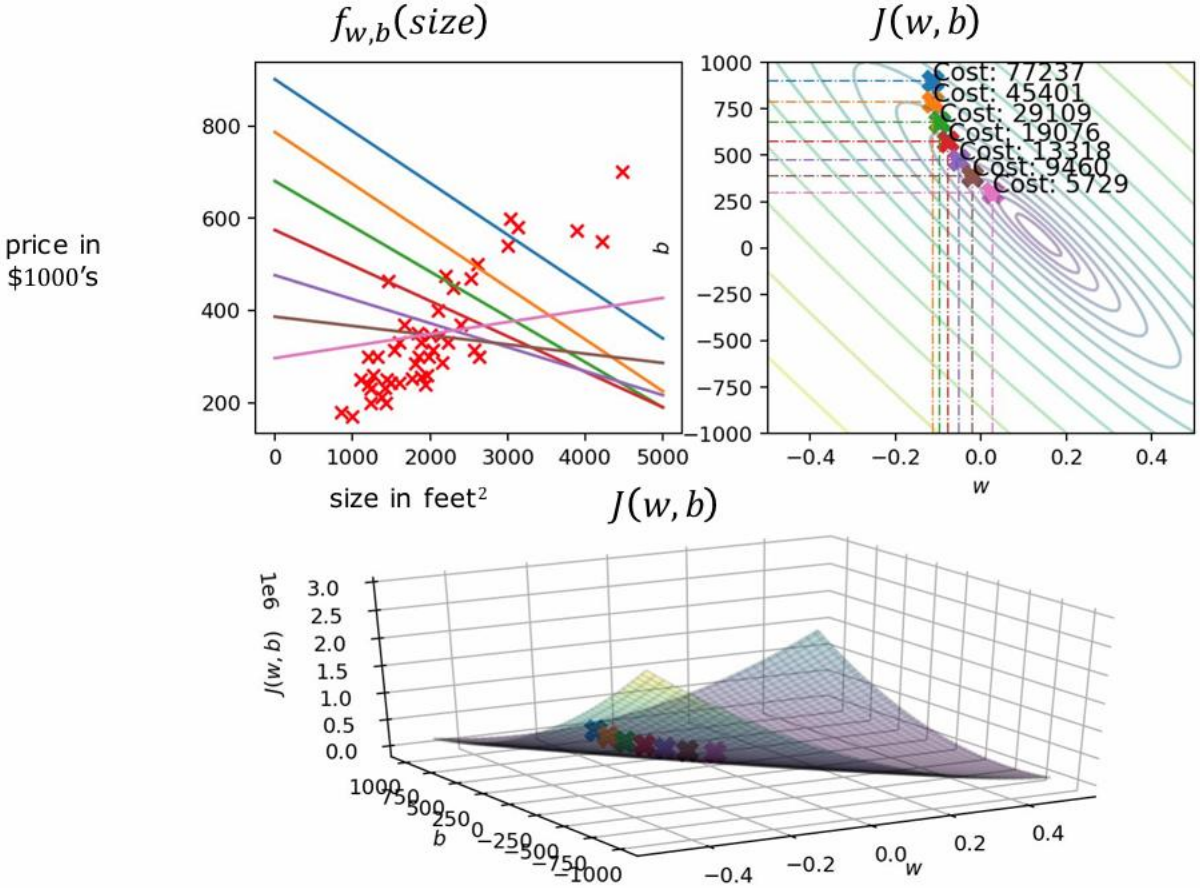
$f_{w,b}(size)$
price in $1000's
size in feet^2
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
w
b
$J(w,b)$
1e6
$J(w,b)$



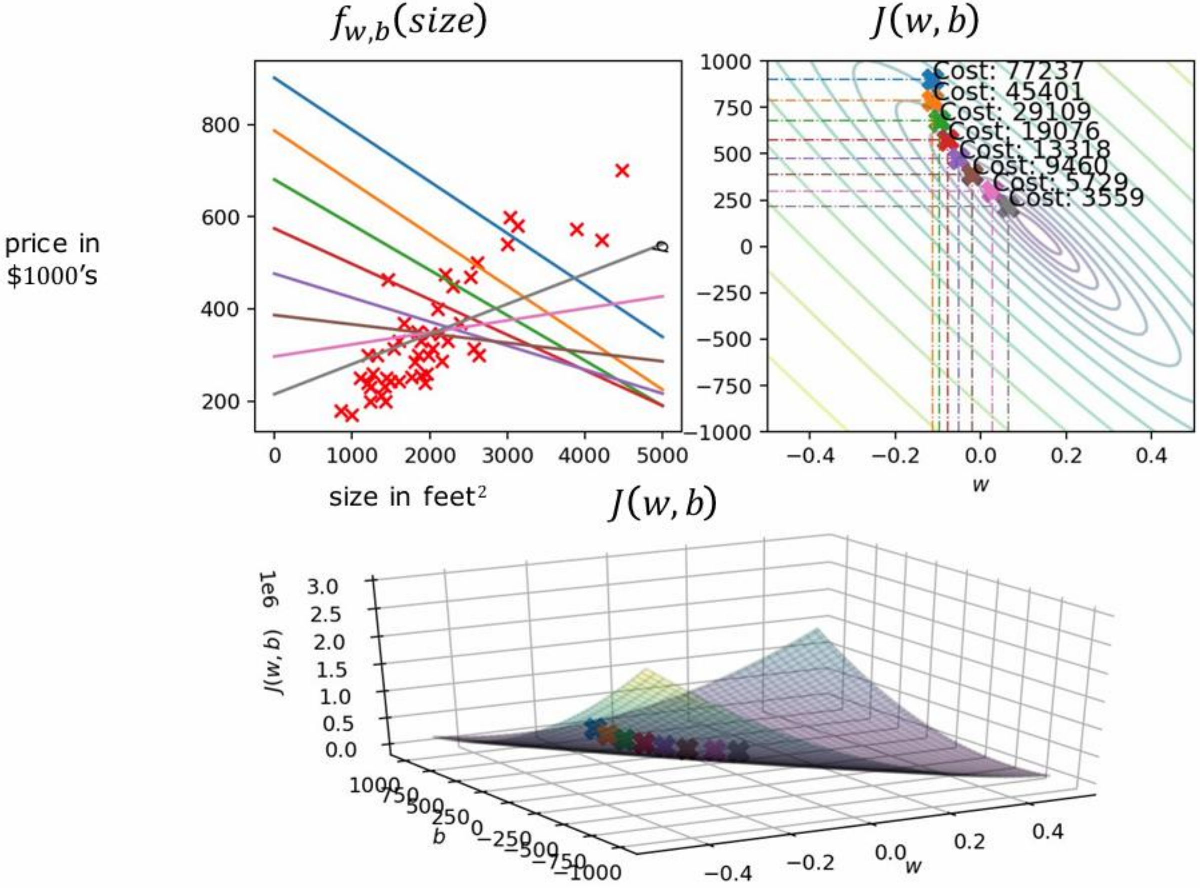
$f_{w,b}(size)$
price in
\$1000's
size in feet^2
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
b
w
$J(w,b)$
1e6
$J(w,b)$
b
w



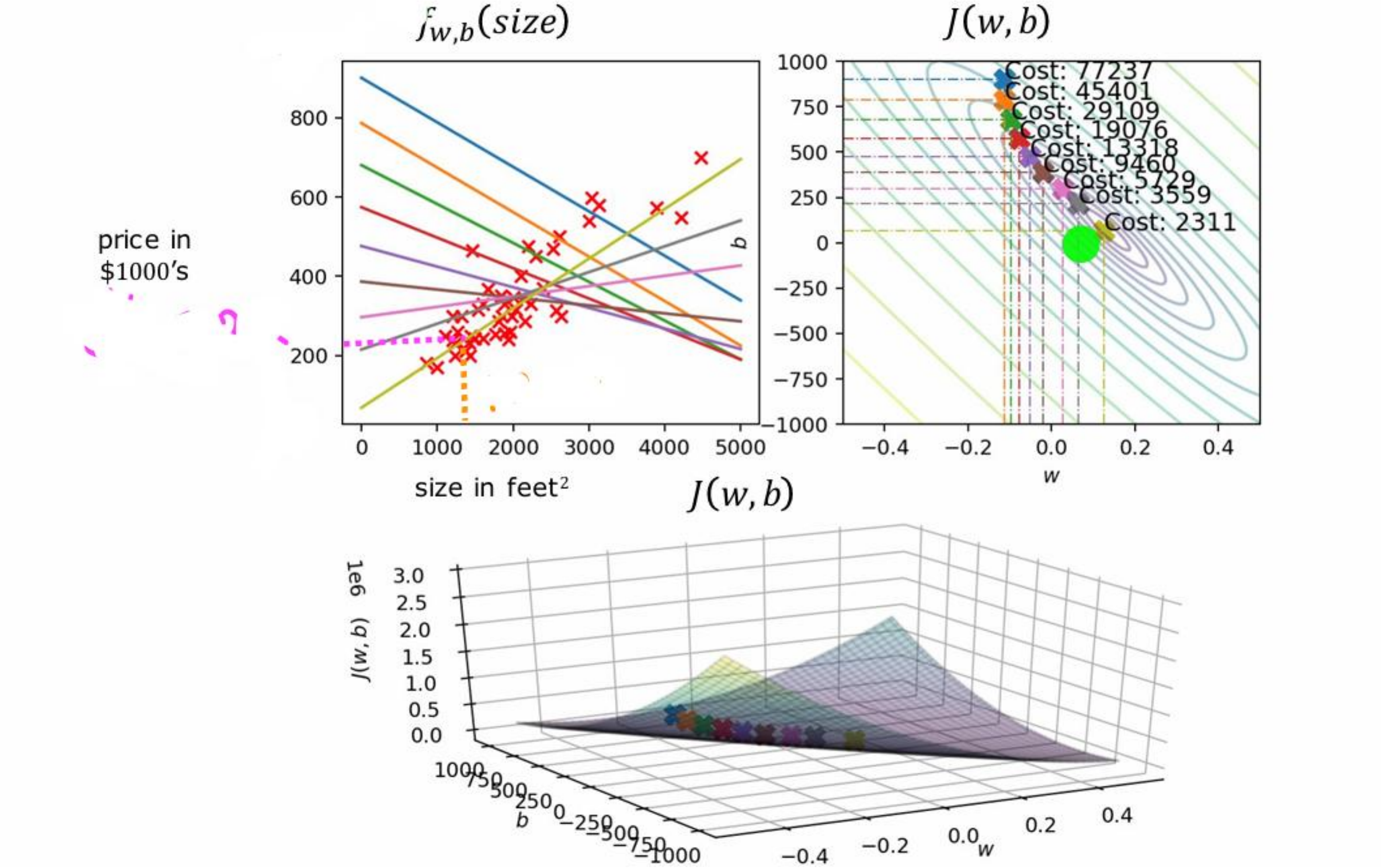
$f_{w,b}(size)$
price in
$1000's
size in feet^2
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
b
w
$J(w,b)$
1e6
J(w, b)
b
w



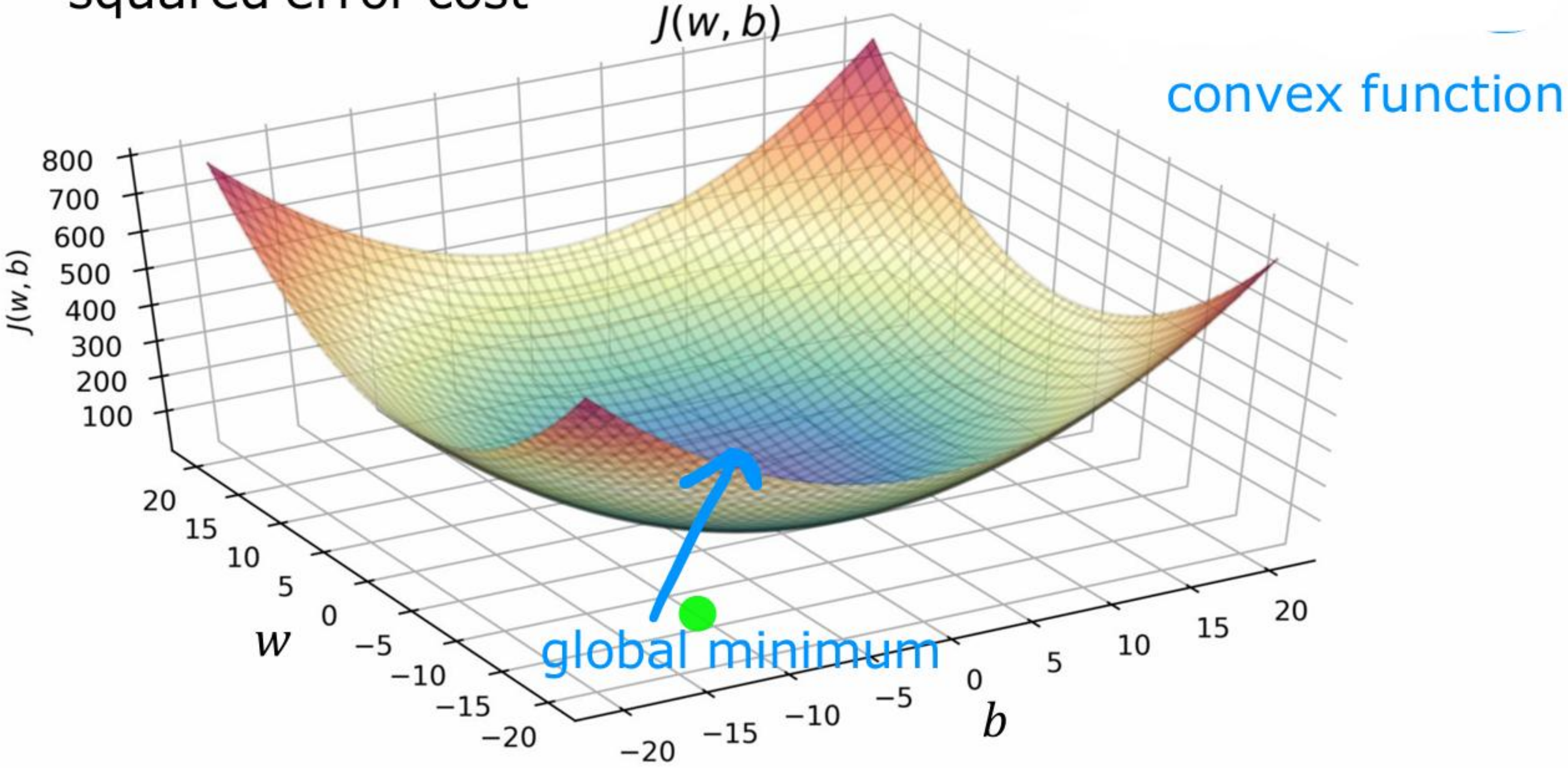
$f_{w,b}(size)$
price in
\$1000's
size in feet^2
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
Cost: 5729
b
w
$J(w,b)$
1e6
$J(w,b)$
b
w



$f_{w,b}(size)$
price in
$1000's
size in feet^2
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
Cost: 5729
Cost: 3559
w
$J(w,b)$
1e6
$J(w,b)$
b
w


$f_{w,b}(size)$
price in
$1000's
size in feet²
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
Cost: 5729
Cost: 3559
Cost: 2311
w
b
$J(w,b)$
1e6
$J(w,b)$
w
b

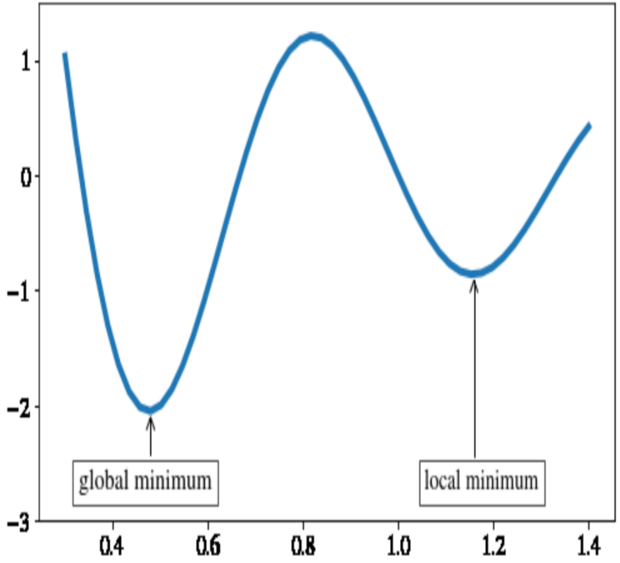
squared error cost



Andrew Ng

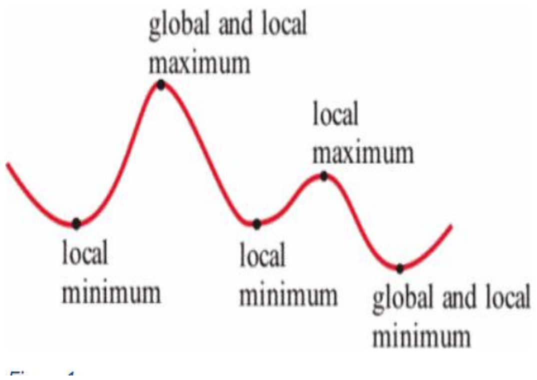
• Global minima and local minima

- **Global Minima:**
- The global minimum is the absolute lowest point of a function over its entire range. It is the most optimal solution, meaning the function cannot go any lower.
- Imagine a bowl-shaped valley—the very bottom of the valley is the global minimum.
- A function has one global minimum (or sometimes none).
- If we reach this point, we have found the best possible solution.



- **Global minima and local minima**

- **Local Minima:**
- A local minimum is a point where the function is lower than nearby points but not necessarily the lowest overall.
- Think of a mountain range with multiple valleys—some valleys are deeper than others. Each of these valleys represents a local minimum, but only the deepest one is the global minimum.
- A function can have multiple local minima.
- If a model gets stuck in a local minimum, it may not reach the best solution.



Conclusion

In today’s lecture, we introduced *Linear Regression* by discussing its purpose, applications, and working principles. We explored the linear equation, cost function, and gradient descent process and wrapped up with a practical Python example.

Key Takeaways:

- Linear regression is a foundational model in machine learning, particularly suited for predicting continuous values.
- The cost function (MSE) helps us assess how well our model fits the data.
- Gradient descent is an optimization algorithm that finds the best parameters for our model.

In our next lecture:

- we’ll build on this by exploring *multivariable linear regression*, where multiple input features influence the prediction. This extension opens up broader applications and prepares us to handle more complex datasets.

Lecture Sources

- **Andrew Ng's Machine Learning Course on Coursera**
- URL: https://www.coursera.org/learn/machine-learning
- **Topics Referenced:**
  - Linear Regression and Gradient Descent
  - Hypothesis and Cost Functions
  - Machine Learning Pipeline and Model Evaluation
- **Book Recommendation:**
  - **"Pattern Recognition and Machine Learning" by Christopher M. Bishop**
    - A comprehensive guide to machine learning cncepts, including probabilistic graphical models and Bayesian methods.
  - **" Machine Learning" by Tom M. Mitchell**
    - A classic and foundational book, often recommended for beginners in machine learning.
  - **" Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville**
    - This is the definitive book for deep learning and is widely used in both academia and industry.
- **Programing Resources**
  - **Python Libraires Documentation**
    - Official Python Documentation: https://docs.python.org/3/
    - Matplotlib Documentation: For creating visualizations.
    - scikit-learn Documentation: For implementing machine learning algorithms.
  - **Online Tutorials**
    - Linear Regression using scikit-learn: A detailed guide with examples.
    - Kaggle Python Tutorials: Great for beginners to understand data science concepts
- My Own Notes and GitHub https://github.com/qazimsajjad/Machine-Learning-Course

Thank you for your attention, and feel free to ask any questions before we wrap up!