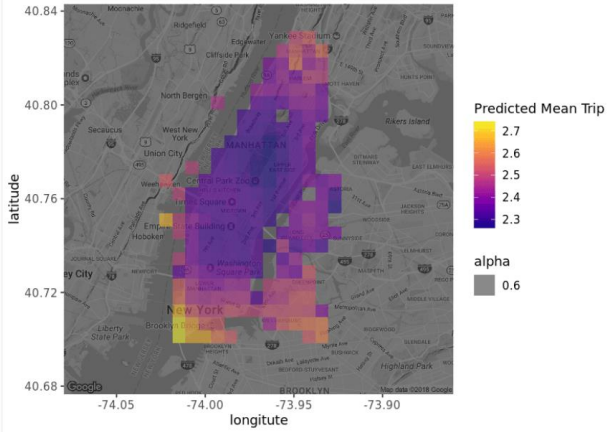
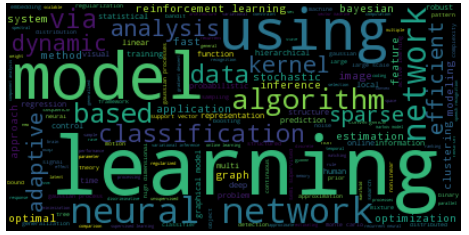
# IT 5504 – Applied Machine Learning

Thushari Silva, PhD
Department of Computational Mathematics
Faculty of Information Technology
University of Moratuwa

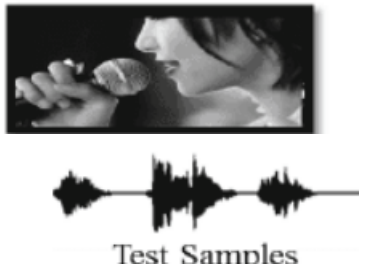
**Predict Taxi Fares with Random Forests**
Predict the location and time to earn the biggest fare for taxi drivers – try Decision Tree and Random Forest drivers

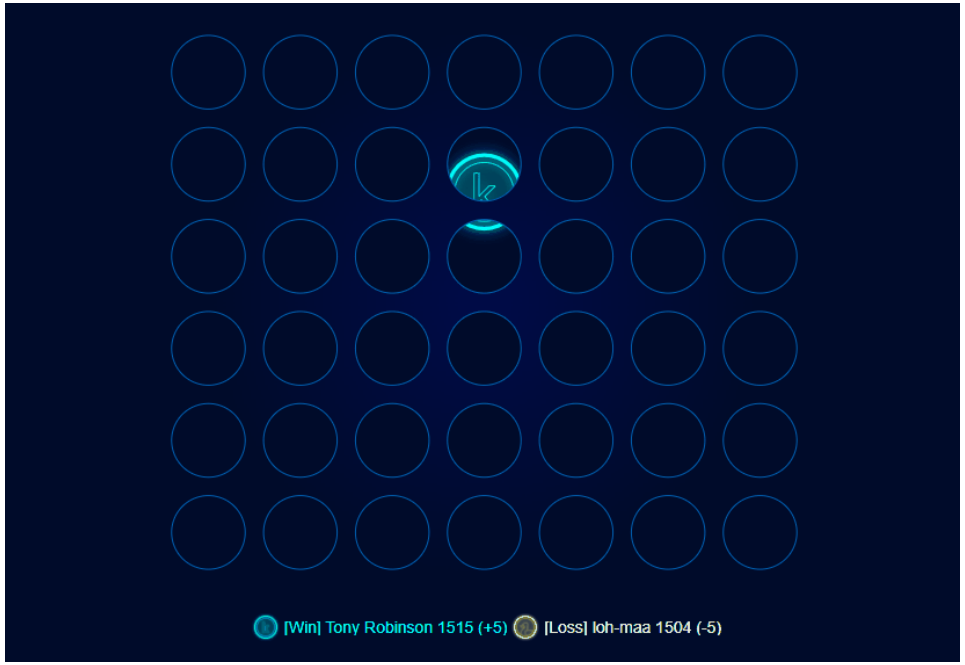


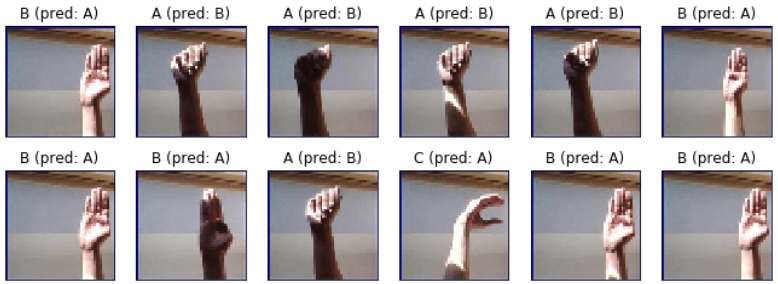

**The Hottest Topics in Machine Learning**
Use text processing and LDA(Linear Discriminant Analysis) to discover the latest trend in machine learning from the large collection of NIPS research papers.

process sound files using Librosa, sound file, and sklearn for the MLPClassifier to recognize emotion from sound files.

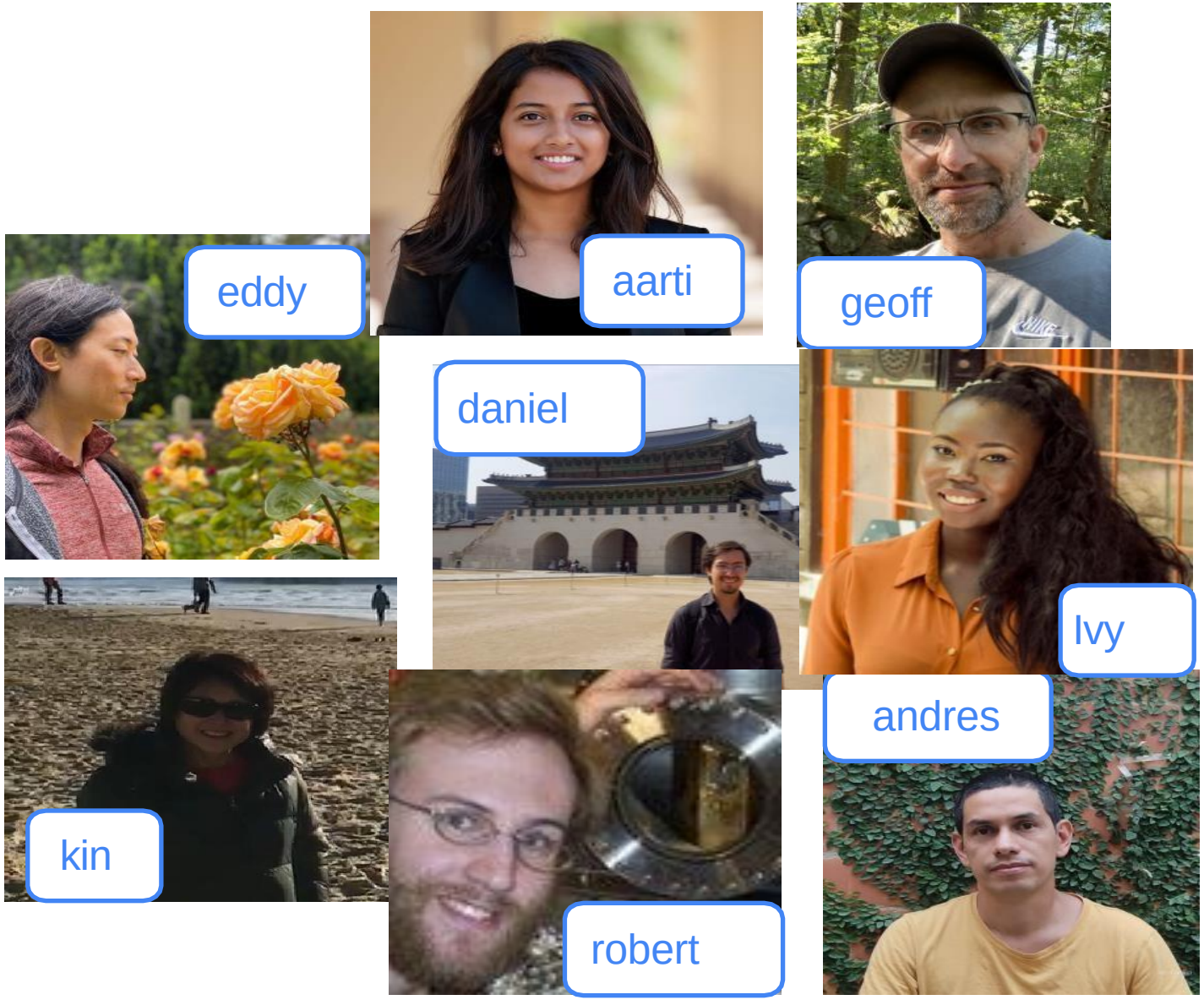


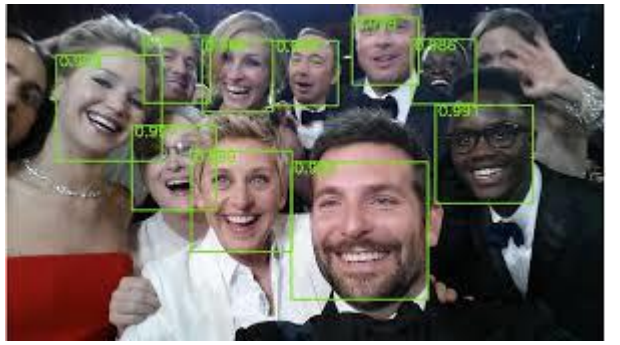




RL (Reinforcement Learning) agent to compete against other Kaggle competition participants.




**American Sign Language image classification. -** CNN (Convolutional Neural Network)



Face Detection

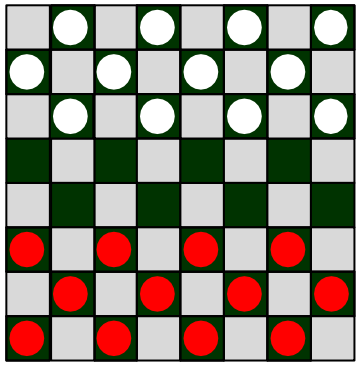
Autonomous driving object detection

# Machine Learning Overview

- What is Machine Learning ?

Machine learning
"Field of study that gives computers the ability to learn without being explicitly programmed."

Arthur Samuel (1959)

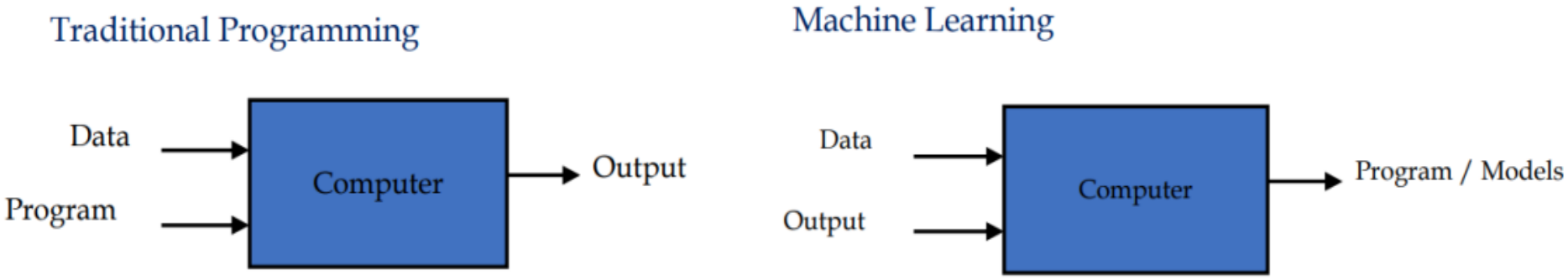


**Tom Mitchell (1997):** *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E."*

This definition formalizes the idea that learning involves improving performance on a task based on experience.

# What is Machine Learning?

- Automating automation
- Getting computers to program themselves
- Writing software is the bottleneck
- Let the data do the work instead




ML defines methods and algorithms that can automatically detect patterns in data and then use these uncovered patterns to predict future data or to perform other kinds of decision-making under uncertainty.

# Question

If the checker's program had been allowed to play only ten games (instead of tens of thousands) against itself, a much smaller number of games, how would this have affected its performance?

- Would have made it better
- Would have made it worse

# Machine learning algorithms

- Supervised learning
  - Classification (e.g., spam detection), Regression (e.g., predicting house prices).
  - **Common Algorithms:** Linear Regression, Logistic Regression, Support Vector Machines (SVM), Decision Trees, Random Forests, Neural Networks.
- Unsupervised learning
  - Clustering (e.g., customer segmentation), Association (e.g., market basket analysis), Dimensionality Reduction (e.g., PCA).
  - **Common Algorithms:** K-Means, Hierarchical Clustering, DBSCAN, Principal Component Analysis (PCA), Autoencoders.
- Recommender systems
- Practical advice for applying learning algorithms

# Machine learning algorithms

- Semi-supervised learning
  - Semi-supervised learning falls between supervised and unsupervised learning.
  - It uses a small amount of labelled data along with a large amount of unlabeled data to improve learning accuracy.
  - Common Algorithms: Self-Training, Co-Training, Generative Models (e.g., GANs used in a semi-supervised context).
- Recommender systems
- Practical advice for applying learning algorithms

# Machine learning algorithms

## Reinforcement Learning:

- An agent interacts with an environment and learns to make decisions by receiving feedback in the form of rewards or penalties. The goal is to maximize cumulative rewards over time.
- **Examples:** Game playing (e.g., AlphaGo), Robotics, Autonomous vehicles.
- **Common Algorithms:** Q-Learning, Deep Q-Networks (DQN), Policy Gradients, Proximal Policy Optimization (PPO).

# Machine learning algorithms

- **Ensemble Learning:**
  - Ensemble learning involves combining multiple models to improve overall performance.
  - The idea is that a group of models working together will outperform any individual model.
  - **Examples:** Random Forests (an ensemble of decision trees), Gradient Boosting Machines (e.g., XGBoost).
  - **Common Techniques:** Bagging, Boosting, Stacking.
- **Online Learning:**
  - Online learning is a method where the model is trained incrementally as new data arrives, rather than on a fixed dataset.
  - This is useful for applications where data is continuously generated.
  - **Examples:** Real-time recommendation systems and adaptive spam filters.
  - **Common Algorithms:** Stochastic Gradient Descent (SGD), Perceptron Algorithm.

# Supervised Learning


Learns from being given “right answers”

# Flower Classification

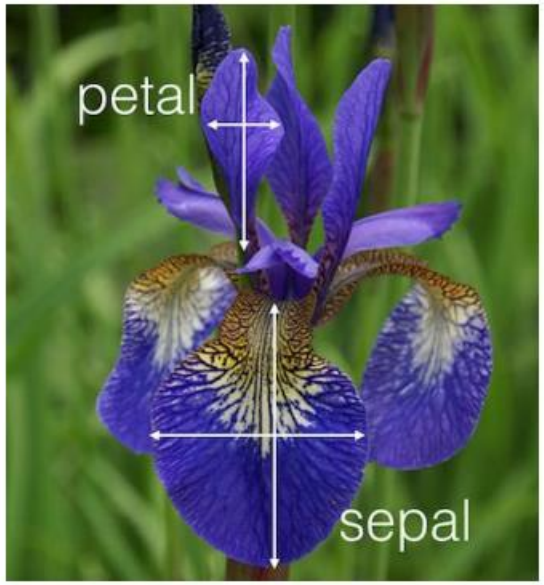
(a)

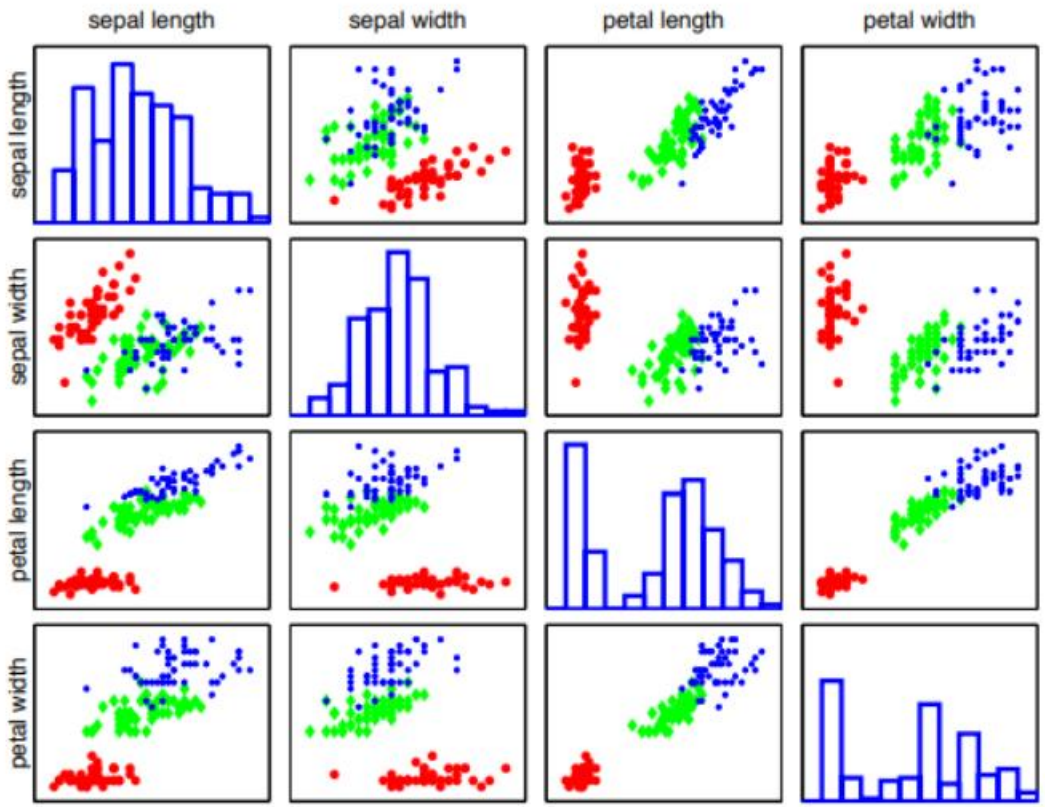
(b)

(c)

Three types of iris flowers: setosa, versicolor and virginica

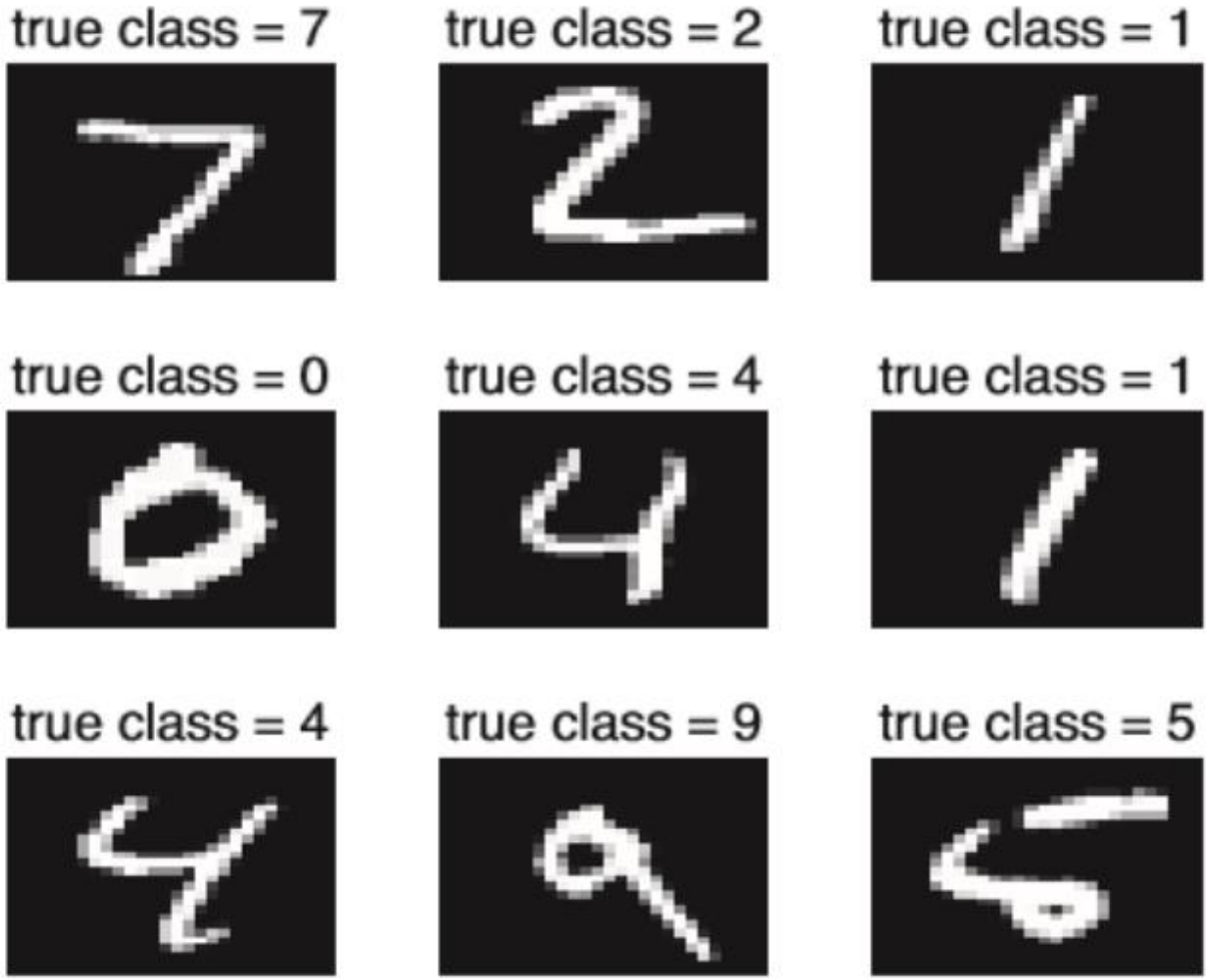
# Flower Classification



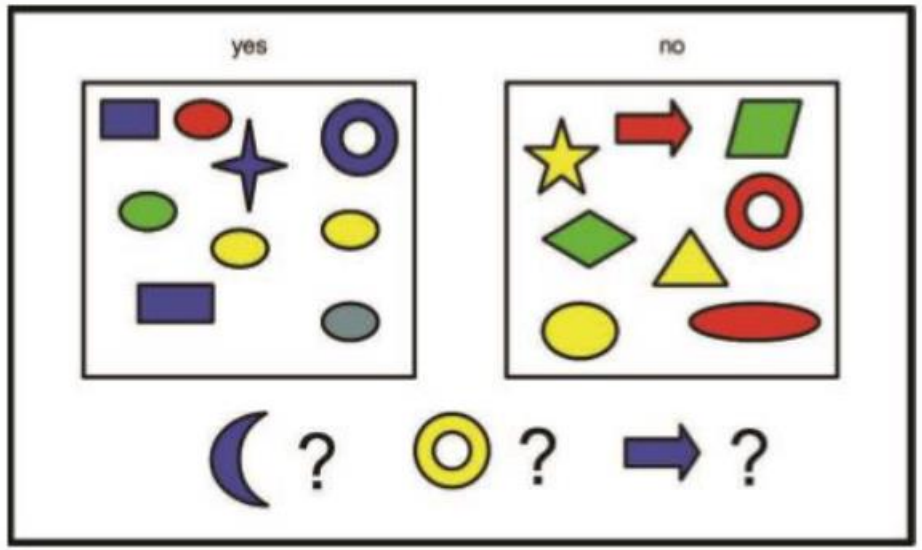




Red circle = setosa, green diamond = versicolor, blue star = virginica.

# Image classification and handwriting recognition

- Supervised learning e.g.




D features (attributes)

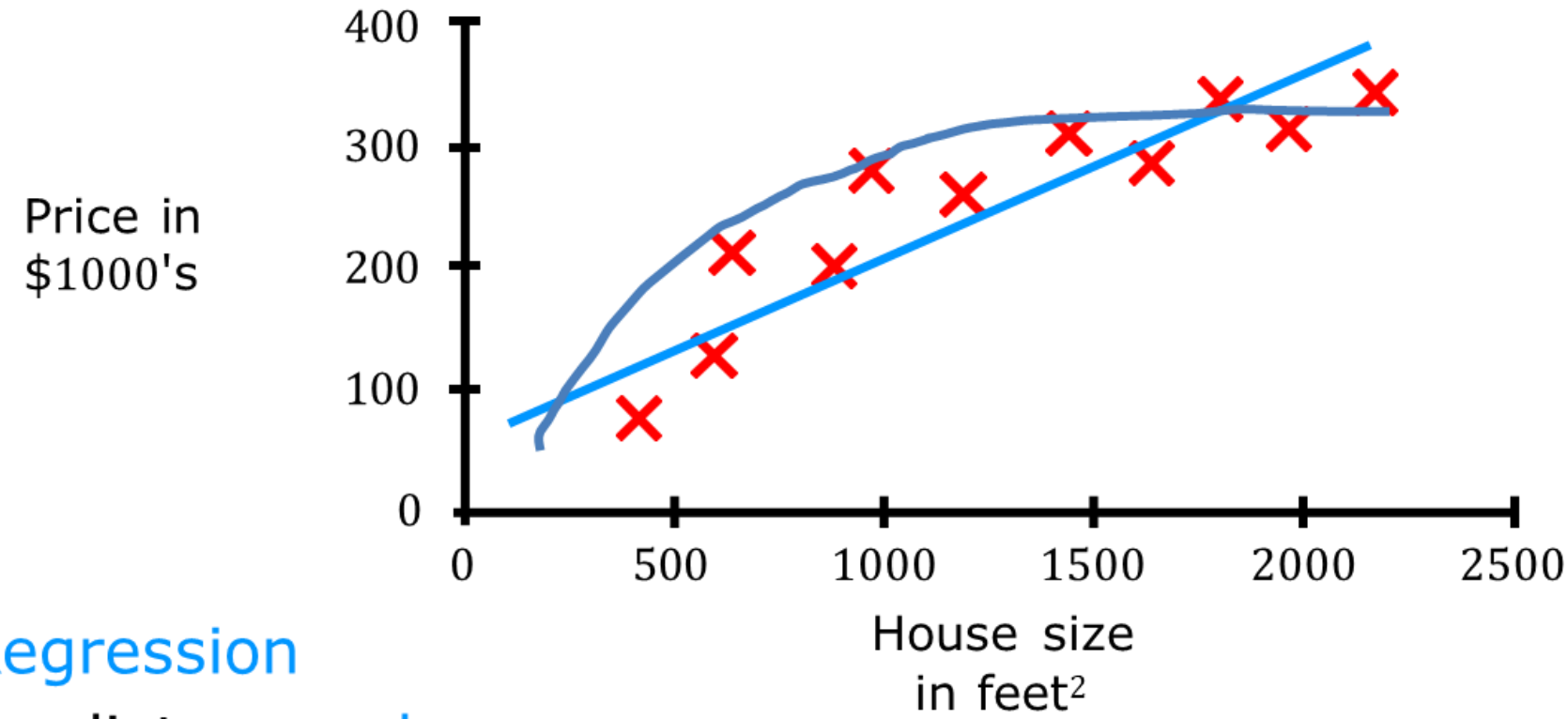
N cases

| Color | Shape | Size (cm) |
|---|---|---|
| Blue | Square | 10 |
| Red | Ellipse | 2.4 |
| Red | Ellipse | 20.7 |

| Label |
|---|
| 1 |
| 1 |
| 0 |

| Input ($X$) | Output ($Y$) | Application |
|---|---|---|
| email | spam? (0/1) | spam filtering |
| audio | text transcripts | speech recognition |
| English | Spanish | machine translation |
| ad, user info | click? (0/1) | online advertising |
| image, radar info | position of other cars | self-driving car |
| image of phone | defect? (0/1) | visual inspection |

# Regression: Housing price prediction

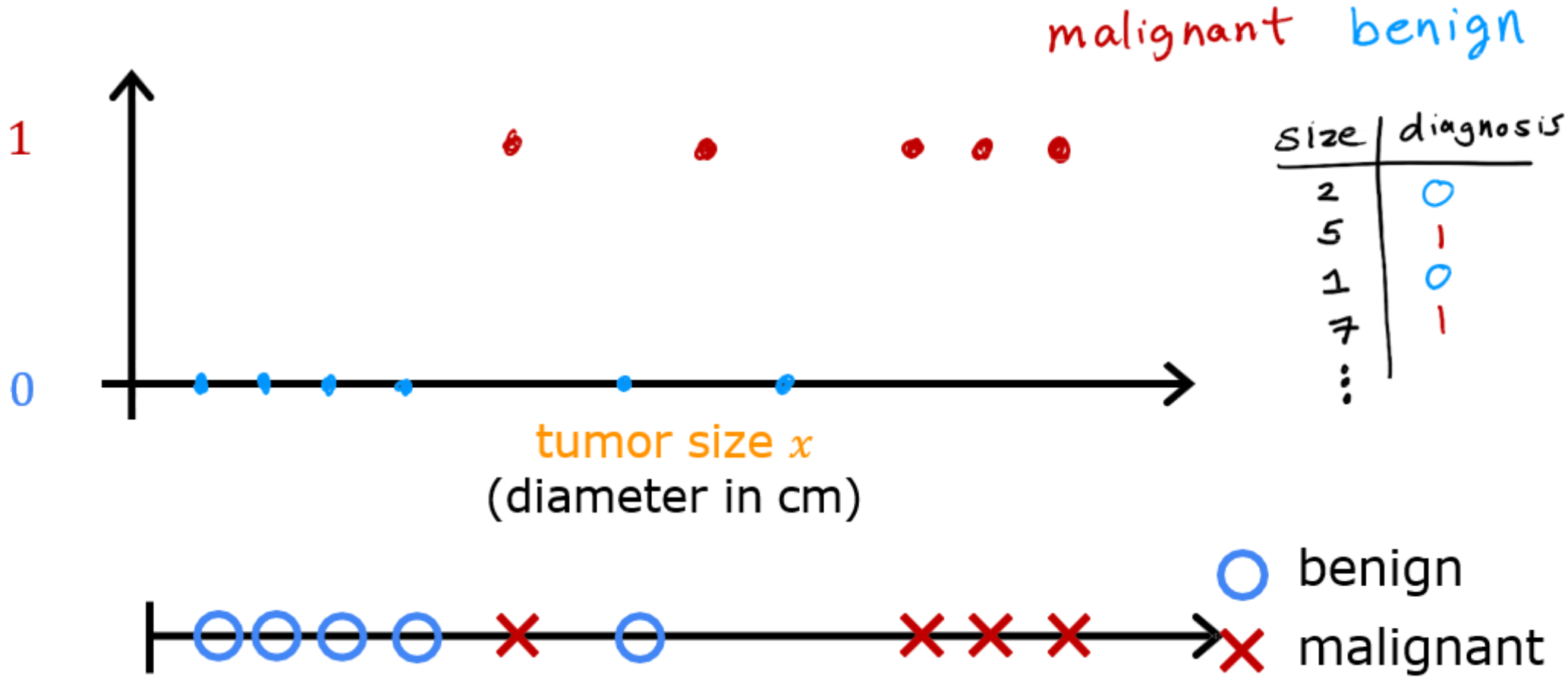



Regression

Predict a number

infinitely many possible outputs

# Classification: Breast cancer detection

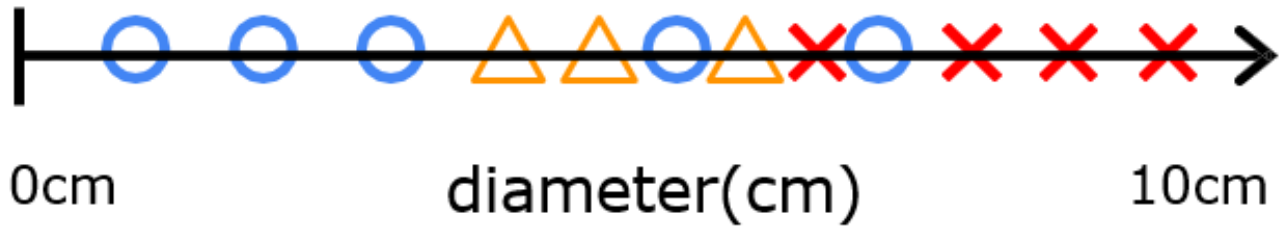


# Classification: Breast cancer detection

benign

malignant type1

malignant type 2

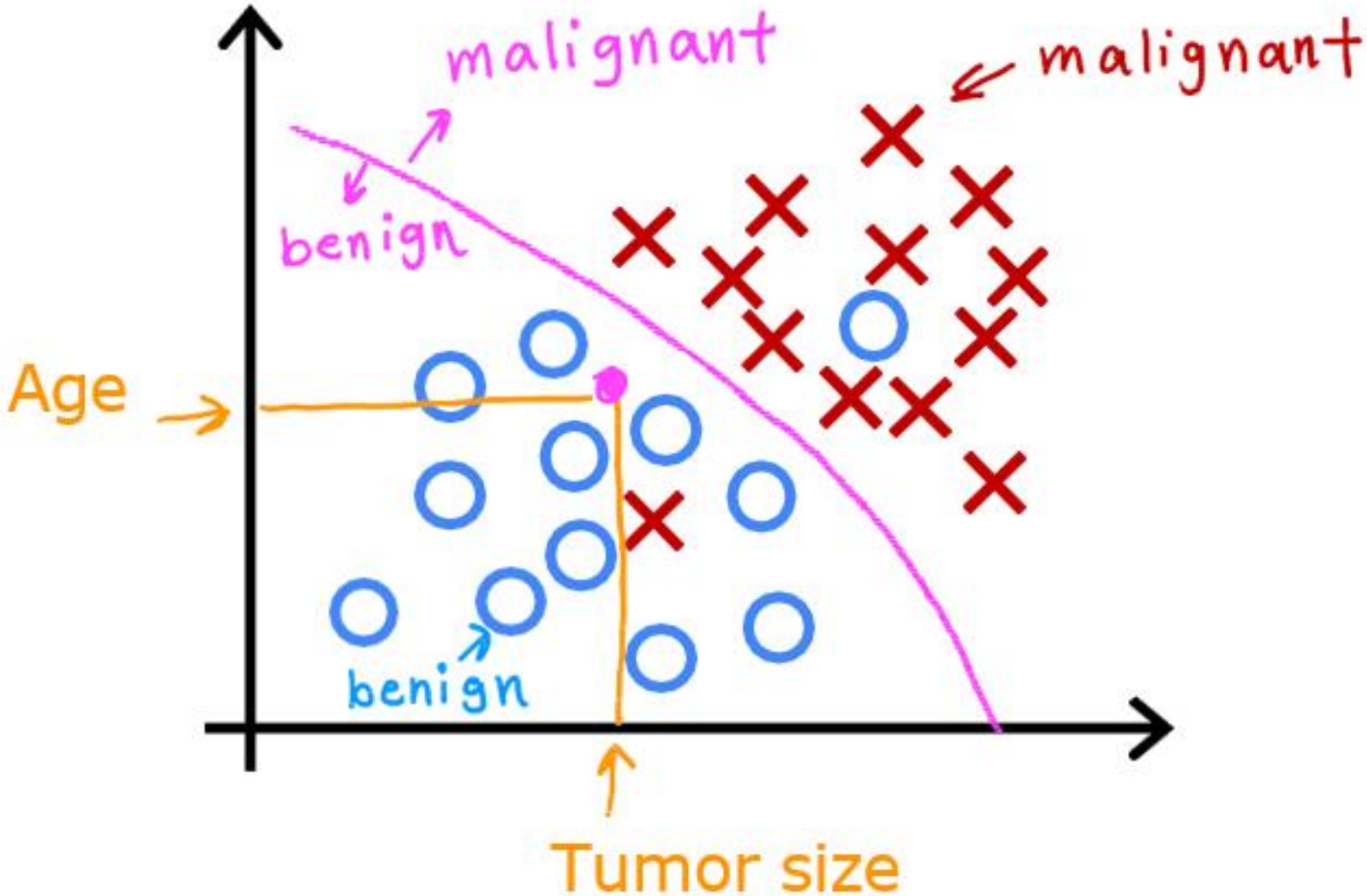



Classification

predict categories

small number of possible outputs

# Two or more inputs



# Supervised learning

Learns from being given "right answers"

Regression

Predict a number

infinitely many possible outputs

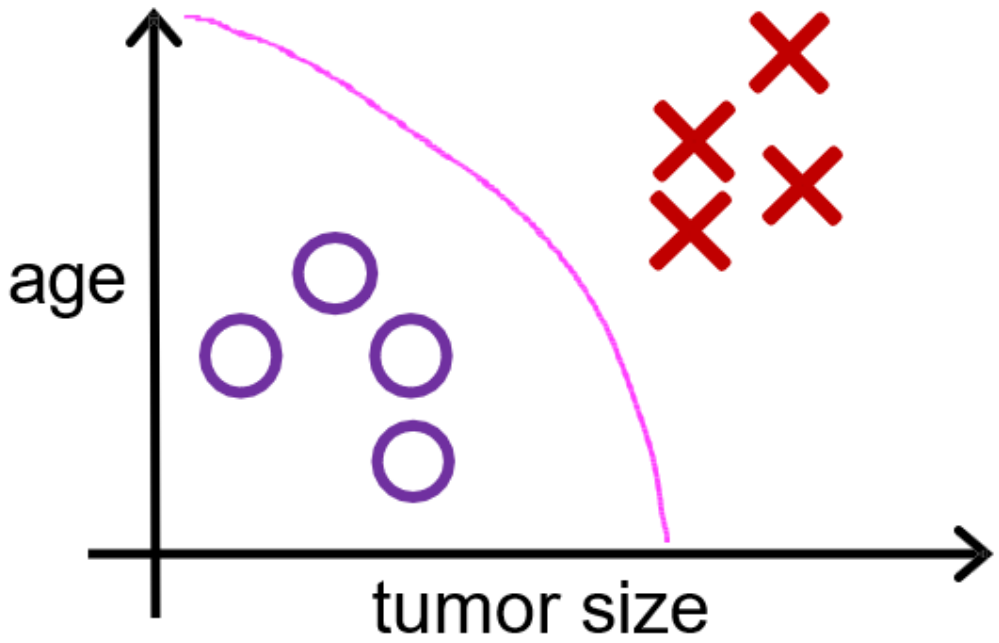
Classification

predict categories

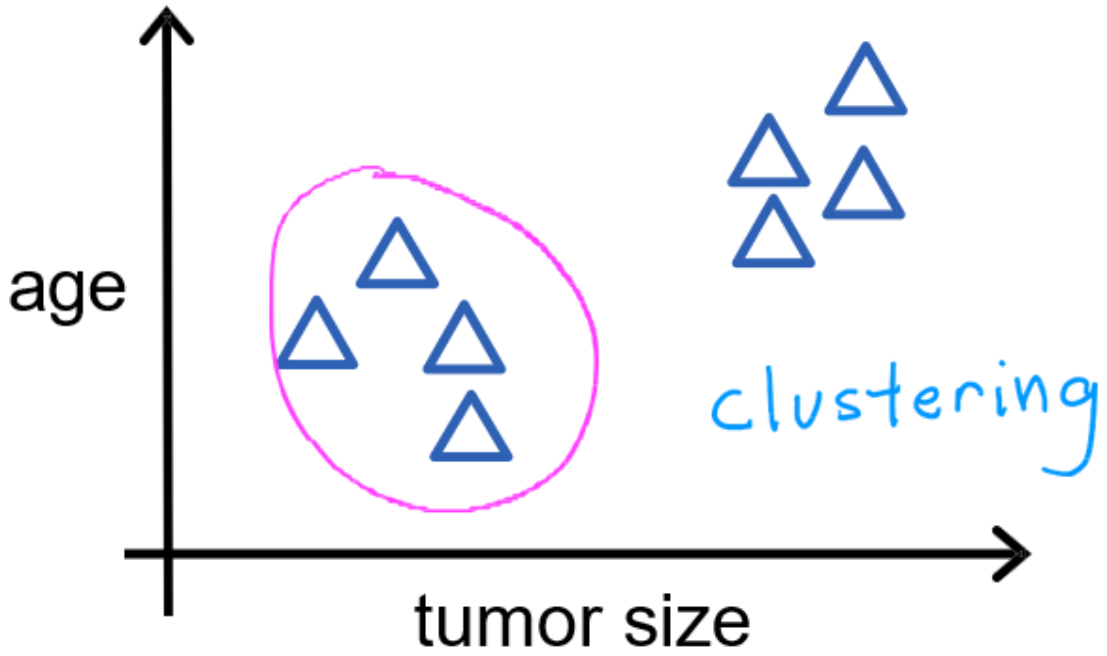
small number of possible outputs

# Unsupervised Learning

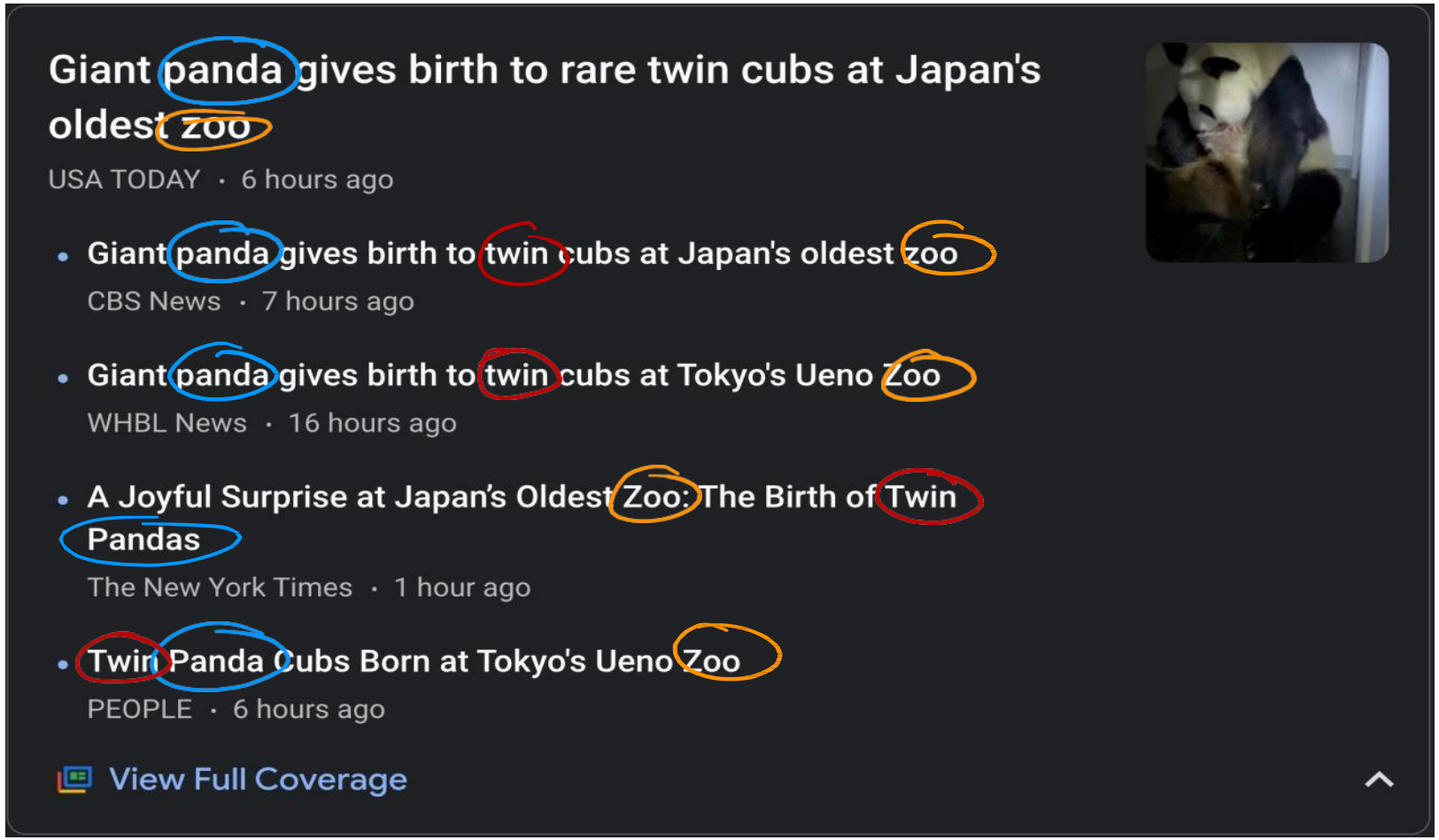
Supervised learning
Learn from data labeled with the "right answers"



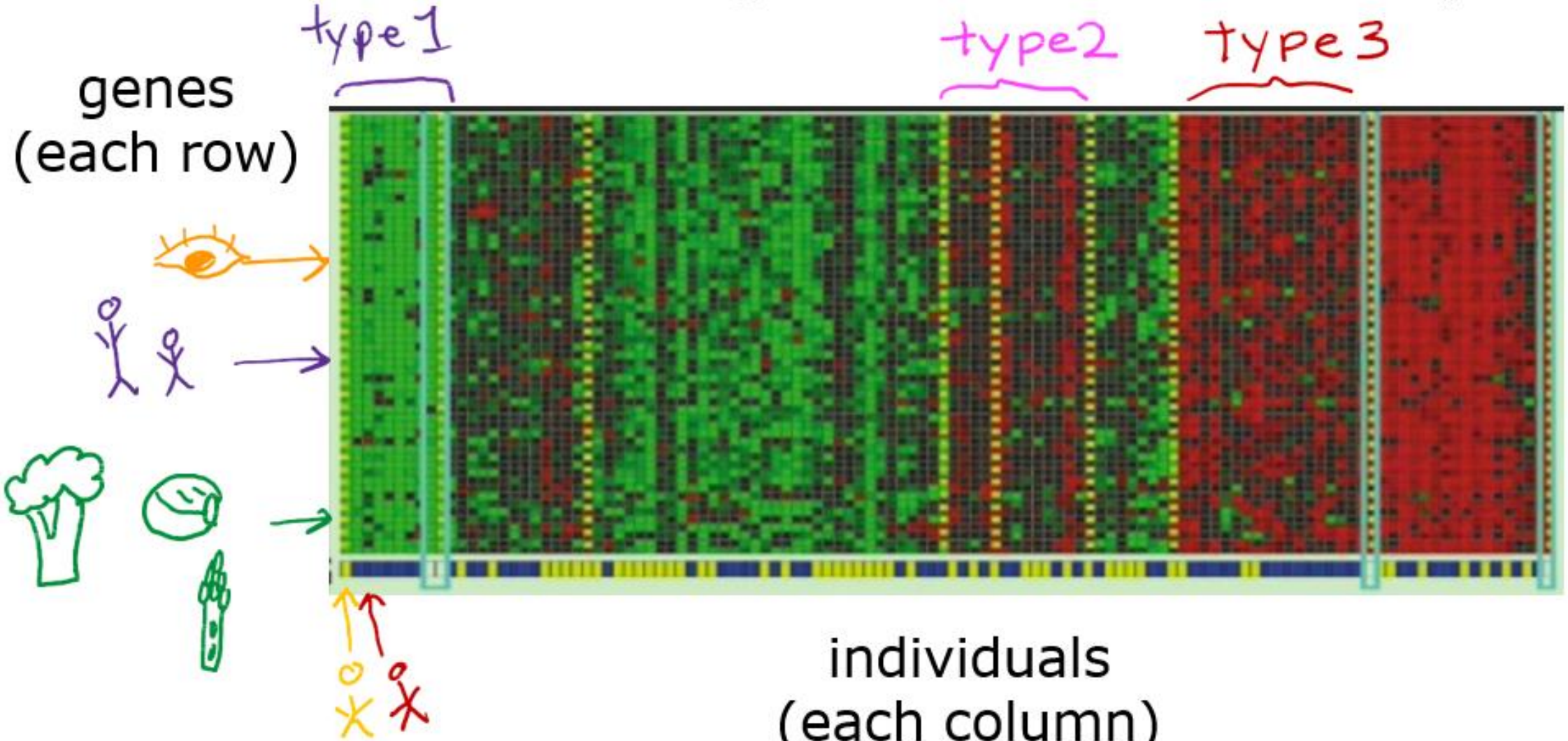

Unsupervised learning
Find something interesting in unlabeled data.



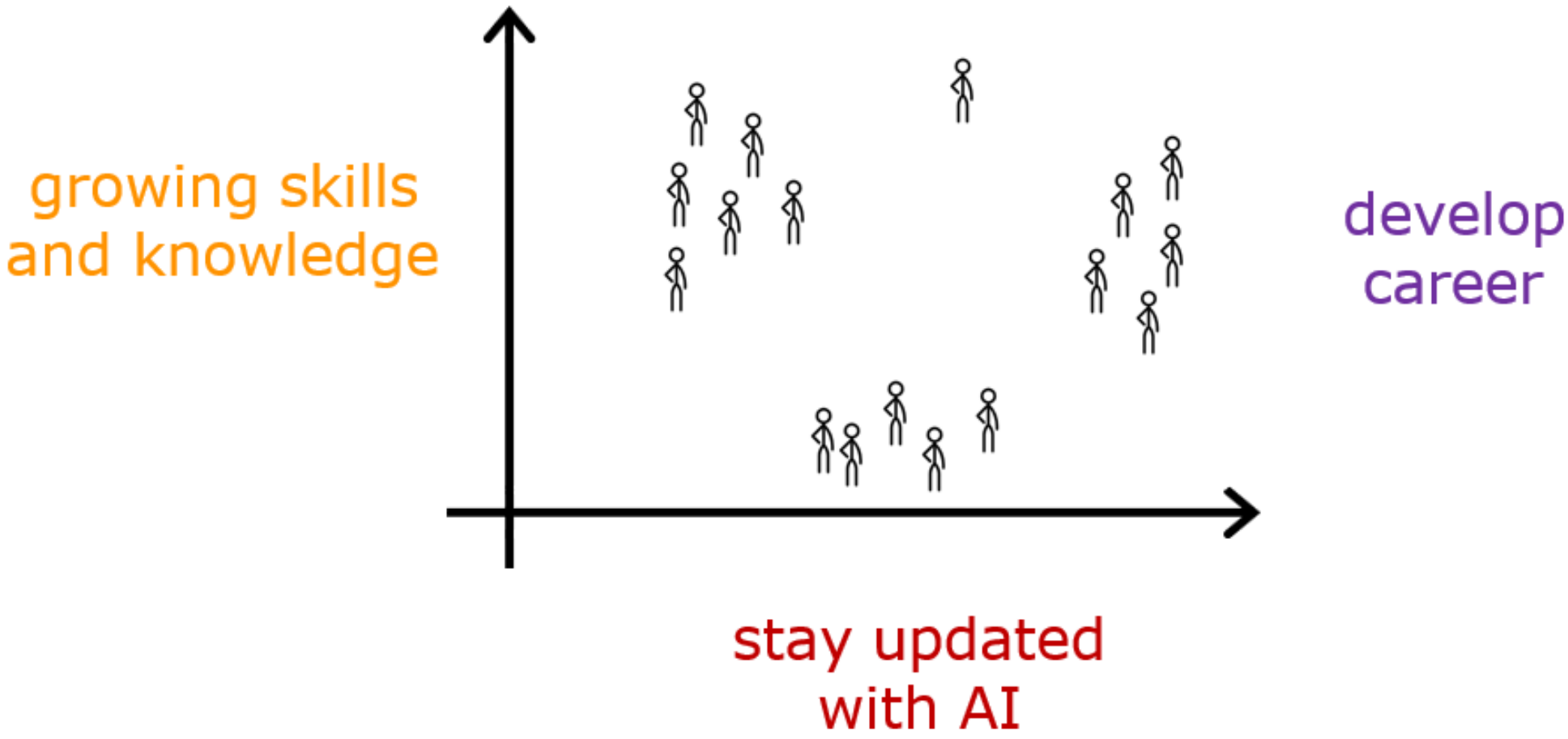
# Clustering: Google news



# Clustering: DNA microarray



# Clustering: Grouping customers



# Unsupervised learning

Data only comes with inputs $x$, but not output labels $y$.
Algorithm has to find structure in the data.

Clustering
Group similar data points together.

Dimensionality reduction
Compress data using fewer numbers.

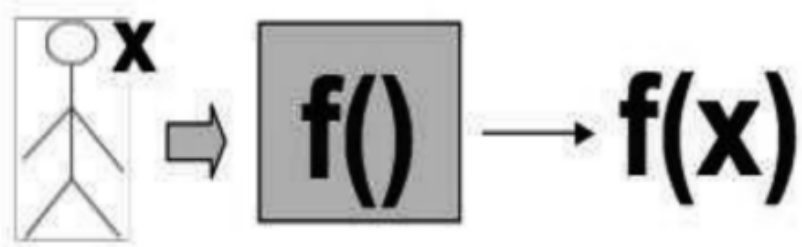
Anomaly detection
Find unusual data points.

# Question

Of the following examples, which would you address using an unsupervised learning algorithm?

- [ ] Given email labeled as spam/not spam, learn a spam filter.
- [ ] Given a set of news articles found on the web, group them into sets of articles about the same story.
- [ ] Given a database of customer data, automatically discover market segments and group customers into different market segments.
- [ ] Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not

# Representing Data




- How do we represent  mathematically?
- Depends on what we're trying to do:
  - deciding to loan money?
  - predicting gender?
- Represent  as a set of attribute-value pairs
  - example: **x** = {height=*180cm*, eyes="*blue*", job="*student*"}

# Attribute-value pairs

- **x** = {height=*180cm*, eyes="*blue*", job="*student*"}
- un-ordered "bag-of-features"
  - if structure is essential – embed it in the attributes
- Have to convert any dataset to this form
- Generally three types of attributes:
  - categorical: *red, blue, brown, yellow*
  - ordinal: *poor, satisfactory, good, excellent*
  - numeric: *-3.14, 6E23, 0, 1*

# Categorical Attributes

- Each instance falls into one of a set of categories
  - **genre**: {*classical, jazz, rock, techno*}
  - categories are mutually exclusive
- Categories usually encoded as numbers
  - no natural ordering to categories
  - only equality testing (=,≠) is meaningful
- Synonymy a major challenge for real datasets:
  - e.g. social tags: *country == folk? house == techno?*

# Ordinal Attribute

- Instance falls into one of a set of categories
- There is a natural ordering to categories
  - **education level**: {*none, school, university, post-graduate*}
  - **Likert scale**: {*disagree, neutral, agree, strongly agree*}
- Encoded as numbers to preserve ordering
  - meaningful to compare values: (<,=,>)
  - should not add / multiply / measure “distance”
- Sometimes hard to differentiate from categorical:
  - does {*single, married, divorced*} have a natural ordering?

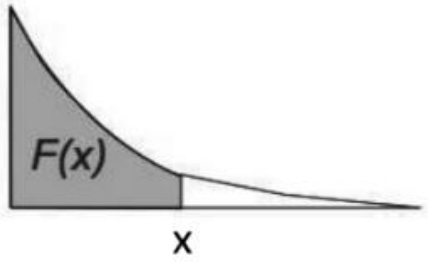
# Numeric Attribute

- Integers or real numbers
  - meaningful to add, multiply, compute mean / variance
  - integers not always the same as real numbers
- Usually want to normalize values (why?)
  - zero mean, unit variance: x' = (x - mean) / st.dev
  - sometimes want [0,1]: x' = (x - min) / (max - min)
- Sensitive to extreme (unusually large/small) values
  - e.g. height: {165,167,171,175,176,181,183,1820}[cm]

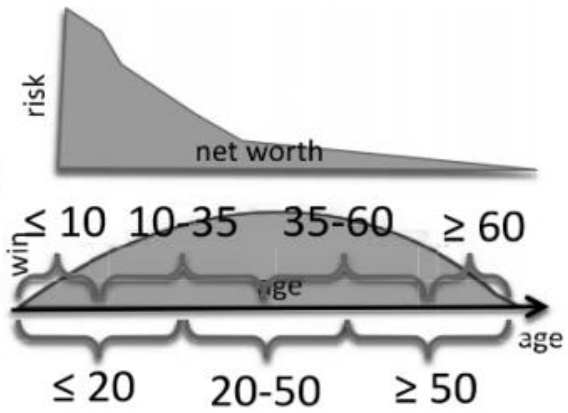
  - must handle this before normalization

# Numeric Attribute Issues

- Skewed distributions
  - systematic extreme values
  - affects regression, kNN, NB; but not DTs
  - simple fix: log(x) or atan(x), then normalize
  - cumulative distribution function: x' = F(x) = P(X≤x)




- Non-monotonic effect of attributes
  - affects regression, NB, DTs(gain); less important for kNN
  - monotonic: net worth and lending risk
    - higher net worth → lower lending risk
  - non-monotonic: age → win a marathon
    - sweet spot: not too young, not too old
  - simple fix: quantization
    - can be unsupervised, overlapping

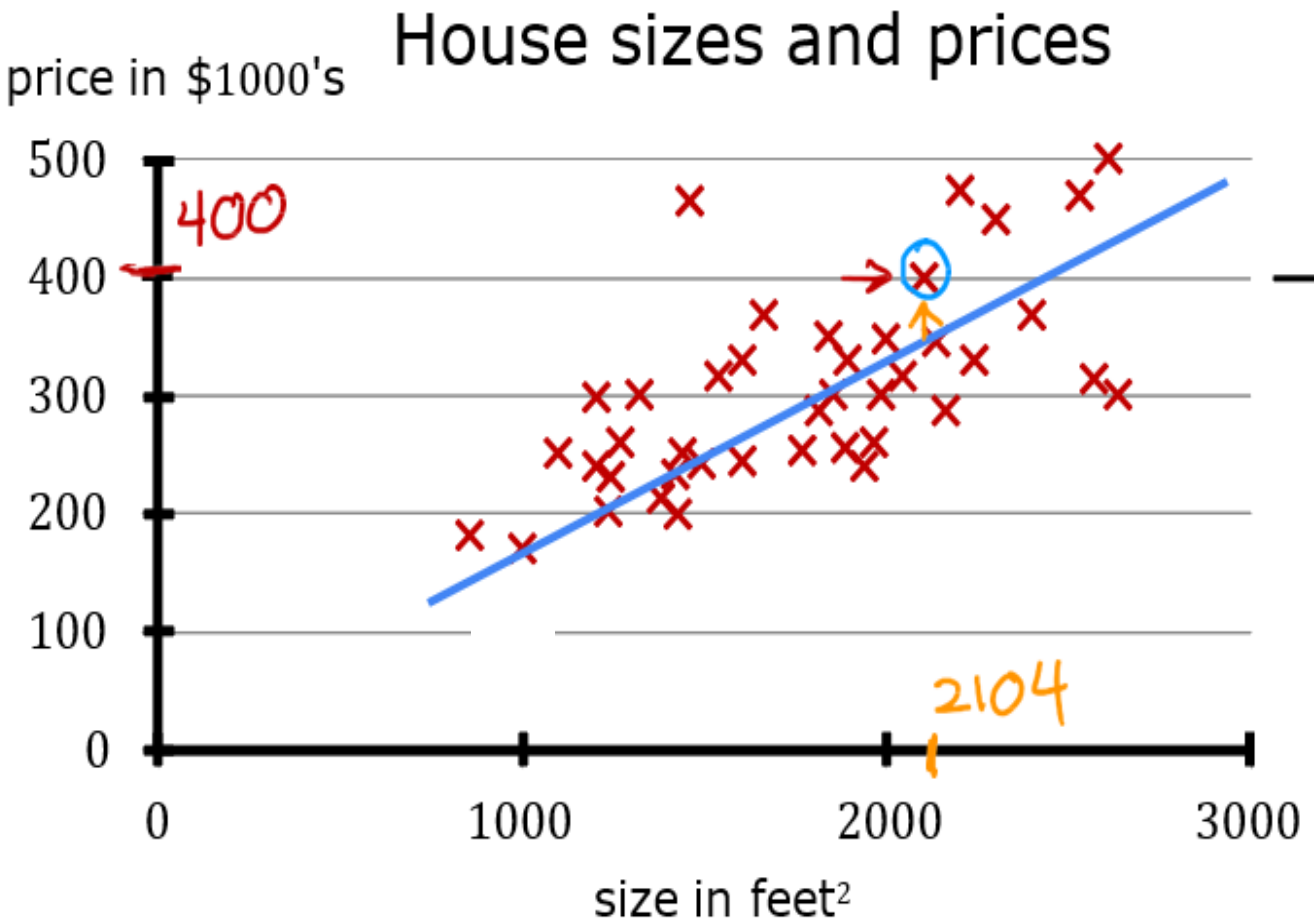


# Linear regression model with one variable


Regression model
Predicts numbers
Infinitely many possible outputs

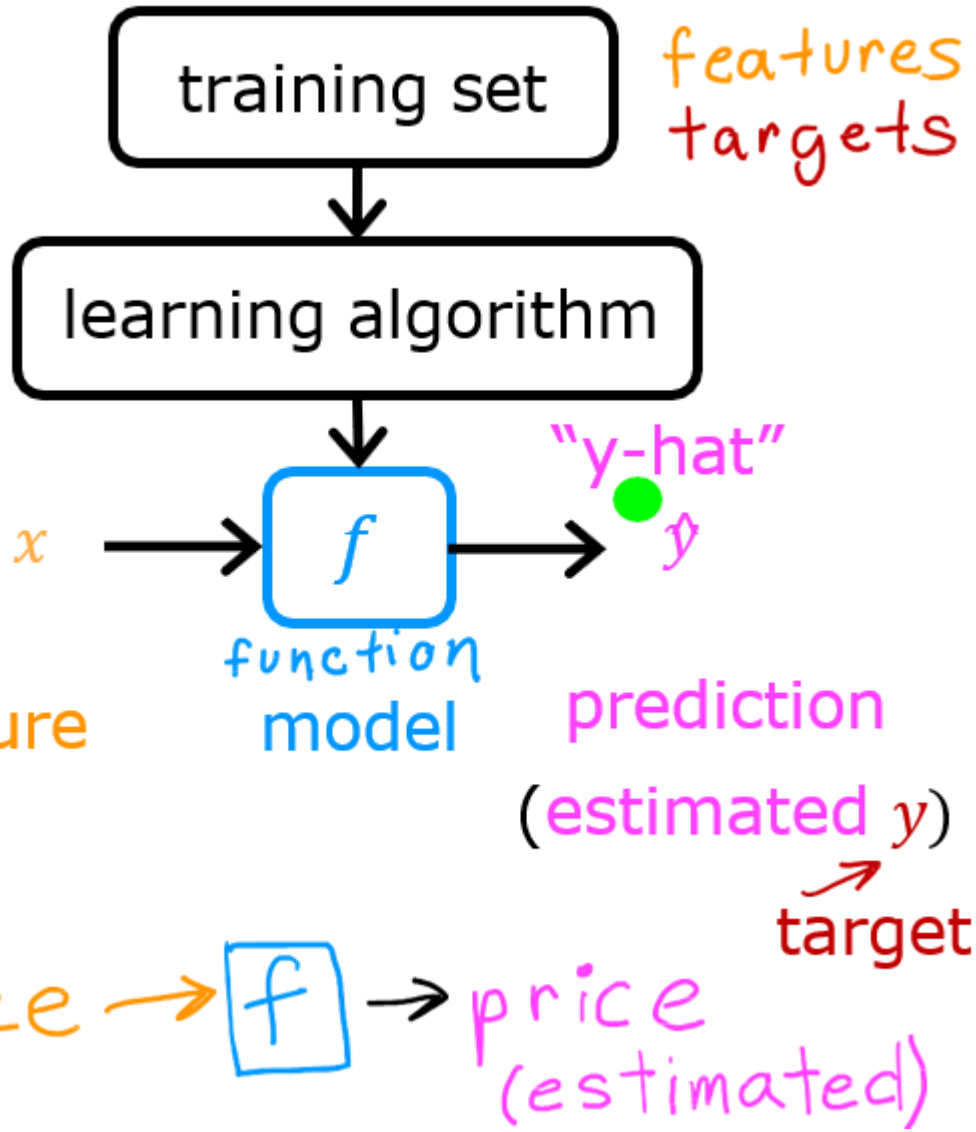
Supervised learning model
Data has "right answers"

Classification model
Predicts categories
Small number of possible outputs




Data table

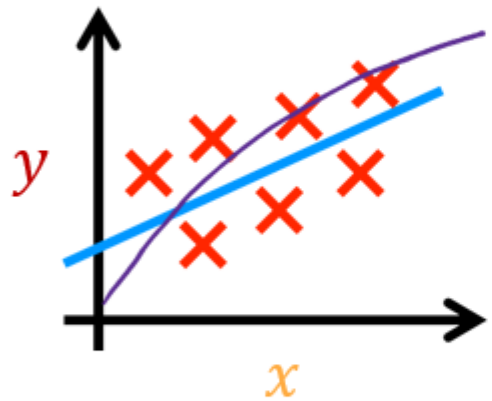
| size in feet² | price in $1000's |
|---|---|
| 2104 | 400 |
| 1416 | 232 |
| 1534 | 315 |
| 852 | 178 |
| ... | ... |
| 3210 | 870 |




# How to represent $f$?

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

$f(x)$




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

$f(x) = wx + b$

linear

single feature x

Linear regression with one variable.

size

Univariate linear regression.

one variable

# Linear Regression with One Variable

## Cost Function

# Training set

features

targets

| size in feet$^2$ ($x$) | price \$1000's ($y$) |
|---|---|
| 2104 | 460 |
| 1416 | 232 |
| 1534 | 315 |
| 852 | 178 |
| ... | ... |

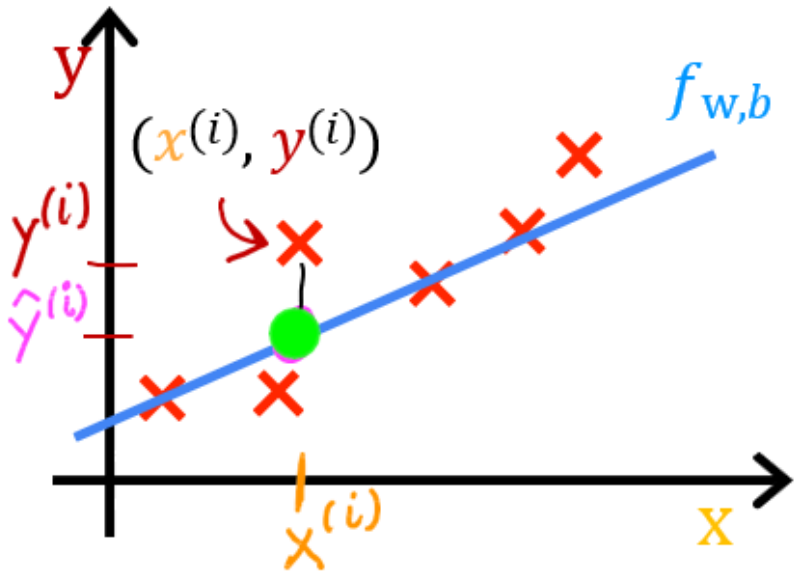
Model: $f_{w,b}(x) = wx + b$

$w, b$: parameters

coefficients

weights

What do $w, b$ do?




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

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

## Cost function: Squared error cost function

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

$\hat{y}^{(i)}$ ↑ error

m = number of training examples

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

$J(w,b)$ ↑ intuition (next!)

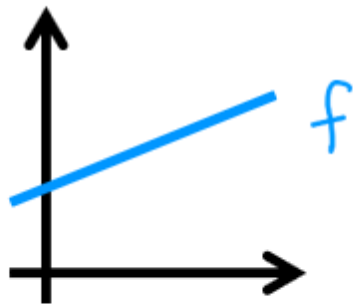
$f_{w,b}$ ↑

Find $w, b$:

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

model:

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




parameters:

$w, b$

cost function:

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

goal:

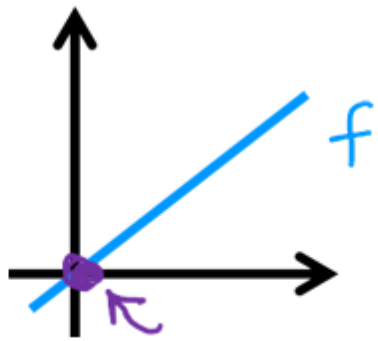
$$\underset{w,b}{\text{minimize}}\, J(w,b)$$

simplified

$$f_w(x) = wx$$

$b = \emptyset$

$w$




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

$wx^{(i)}$

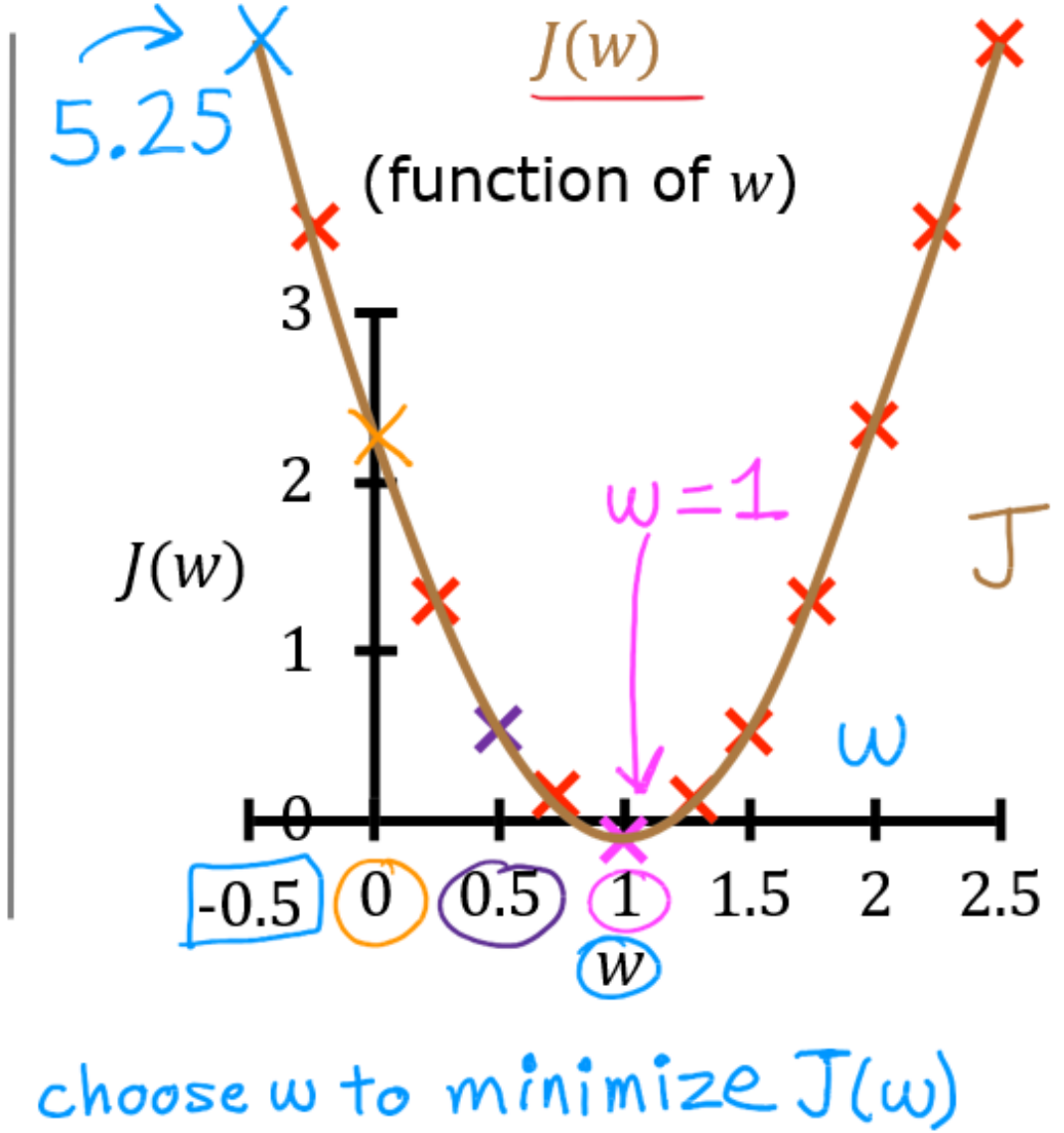
$$\underset{w}{\text{minimize}}\, J(w)$$

goal of linear regression:

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

general case:

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



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

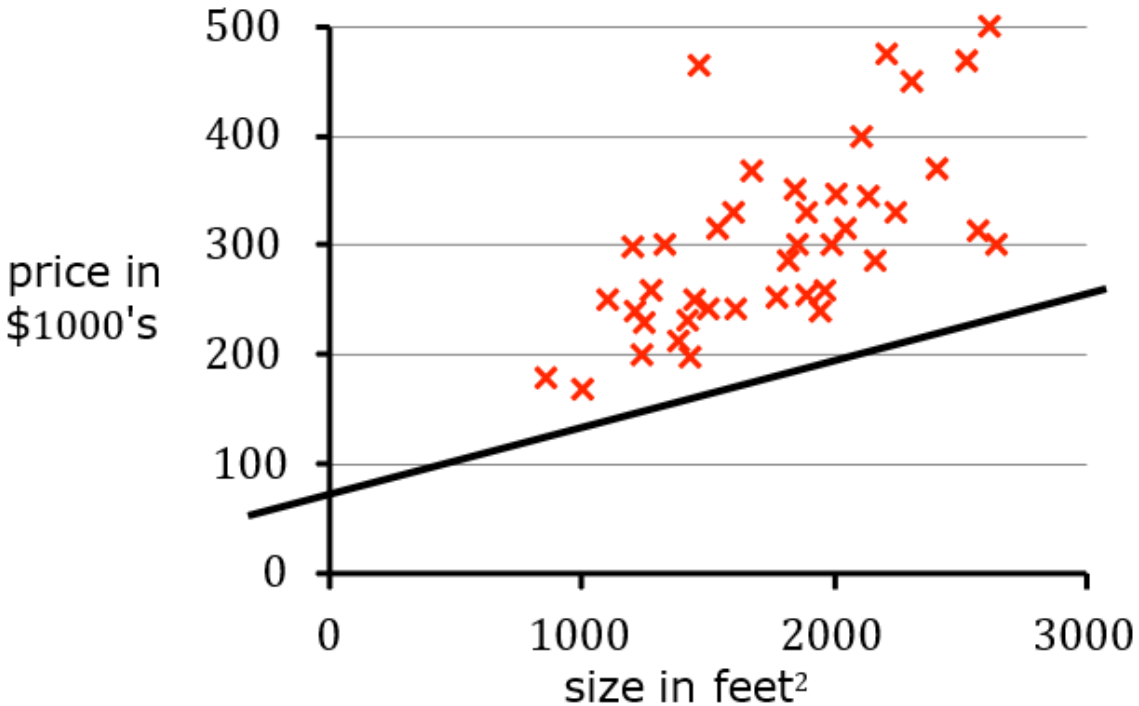
Parameters $w, b$

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

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

$f_{w,b}$

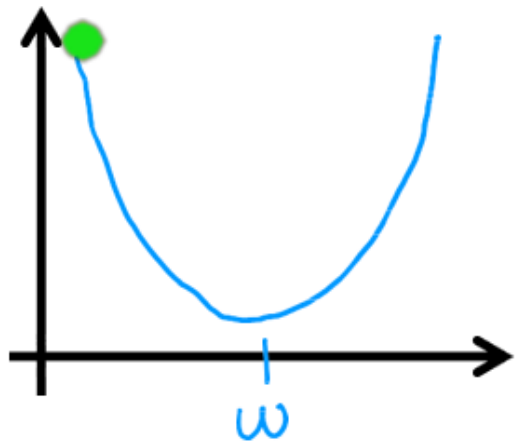
(function of $x$)



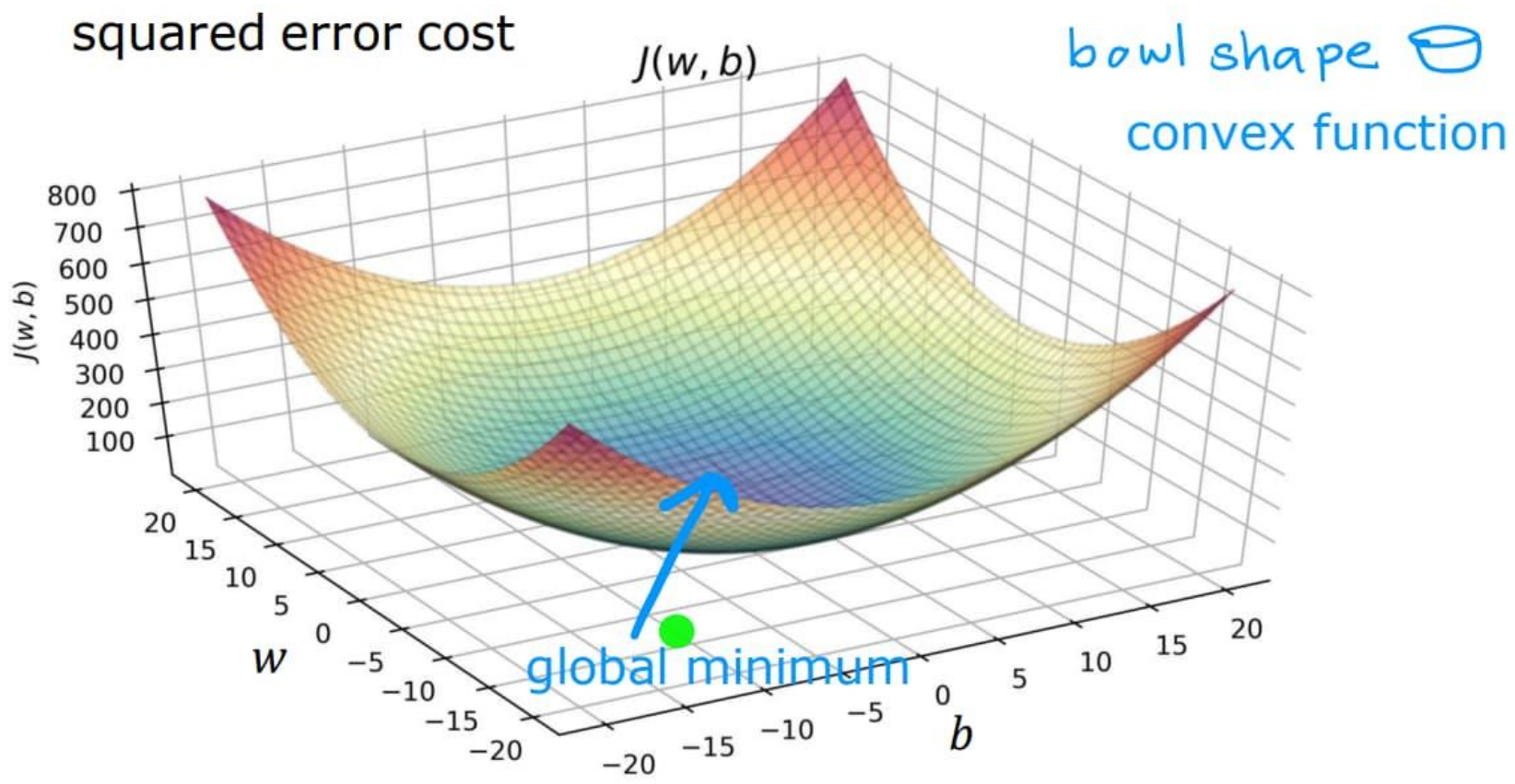

$$f_{w,b}(x) = 0.06x + 50$$

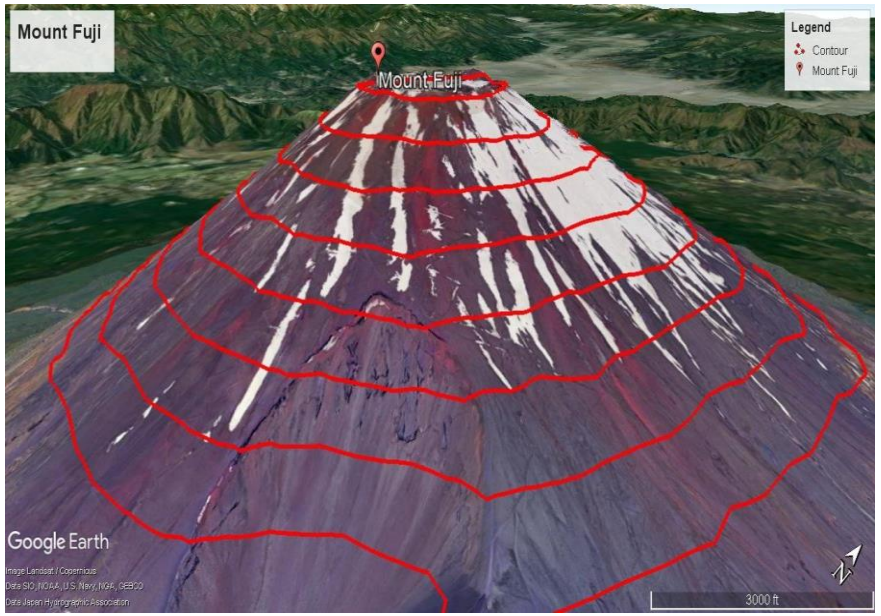
$J$

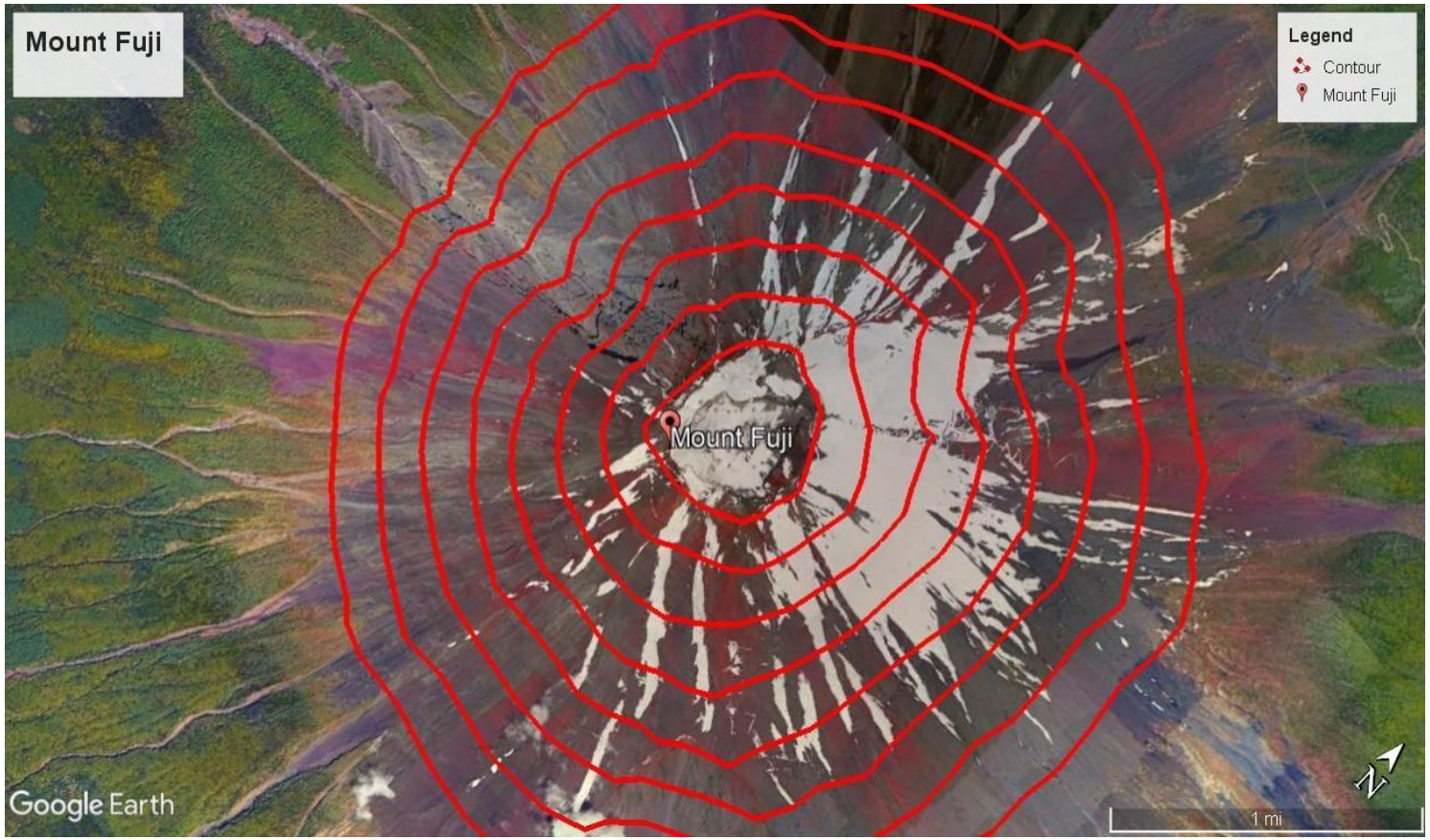
(function of $w, b$)



# Local/Global minimum in cost Function




Mount Fuji
Legend
Contour
Mount Fuji
Mount Fuji
Google Earth
3000 ft


Mount Fuji
Legend
Contour
Mount Fuji
Mount Fuji
Google Earth
1 mi

Have some function $J(w, b)$

for linear regression or any function

Want $\min_{w,b} J(w, b)$

$\min_{w_1, \ldots, w_n, b} J(w_1, w_2, \ldots, w_n, b)$

Outline:

- Start with some $w, b$ (set $w=0$, $b=0$)
- Keep changing $w, b$ to reduce $J(w, b)$
- Until we settle at or near a minimum

J not always

may have >1 minimum

# Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{d}{dw} J(w,b)$$

$$b = b - \alpha \frac{d}{db} J(w,b)$$

Learning rate

Derivative

Simultaneously update w and b

Assignment

$a = c$

$a = a + 1$

Code

Truth assertion

$a = c$

$a = a + 1$ ✗

Math

a==c

## Correct: Simultaneous update

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$$w = tmp\_w$$

$$b = tmp\_b$$

## Incorrect

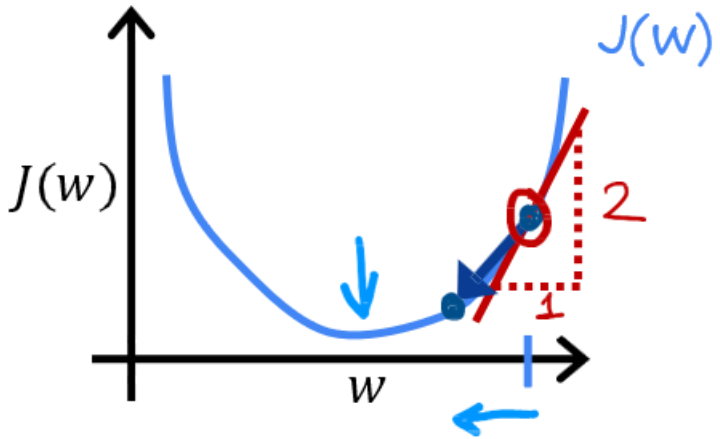
$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

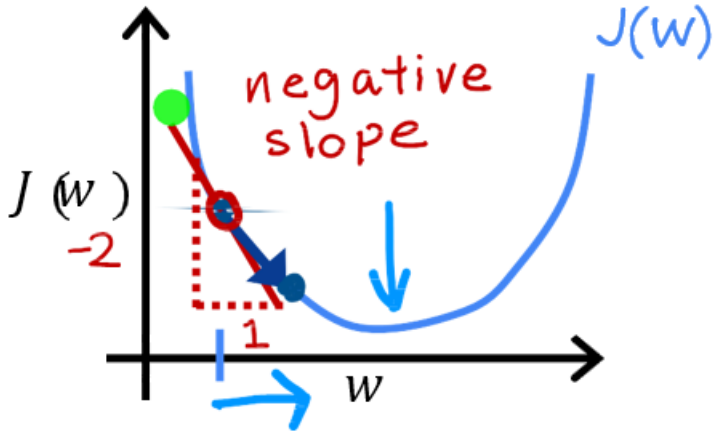
$$w = tmp\_w$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$$b = tmp\_b$$

# Gradient Descent Intuition




$$w = w - \alpha \underbrace{\frac{d}{dw} J(w)}_{>0}$$

$$w = w - \alpha \cdot (positive\ number)$$

$$\underbrace{\frac{d}{dw} J(w)}_{<0}$$

$$w = w - \alpha \cdot (negative\ number)$$

# Learning Rate

$$w = w - \alpha \frac{d}{dw} J(w)$$

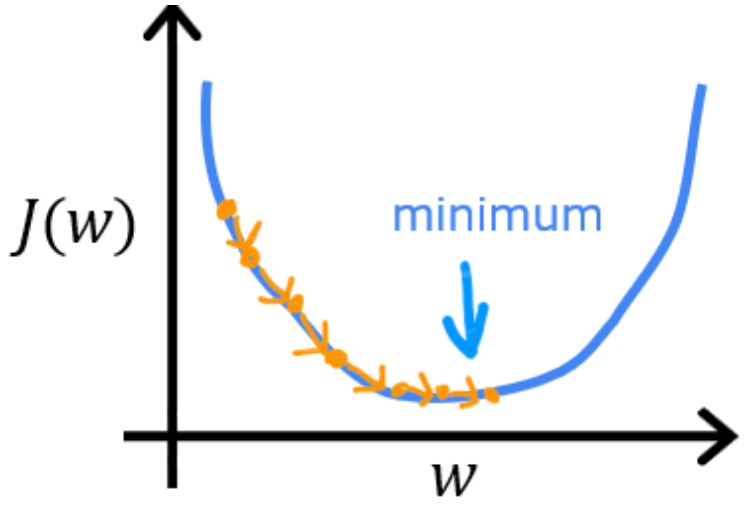
If $\alpha$ is too small...

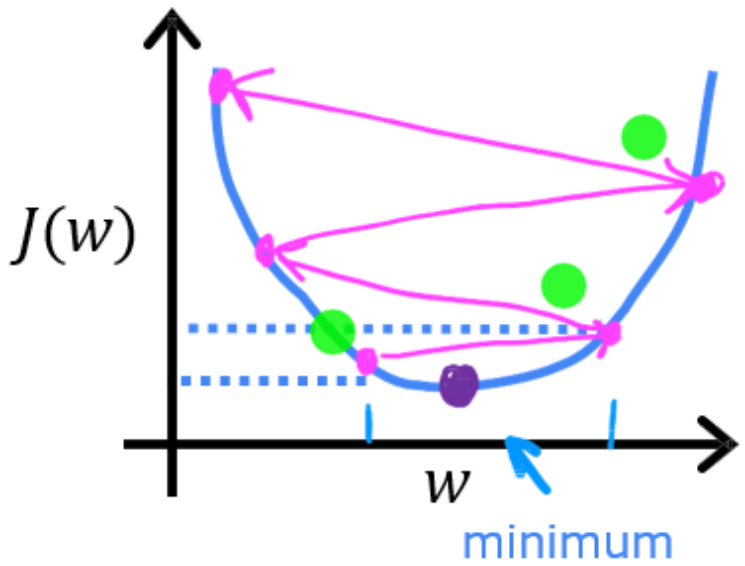
Gradient descent may be slow.

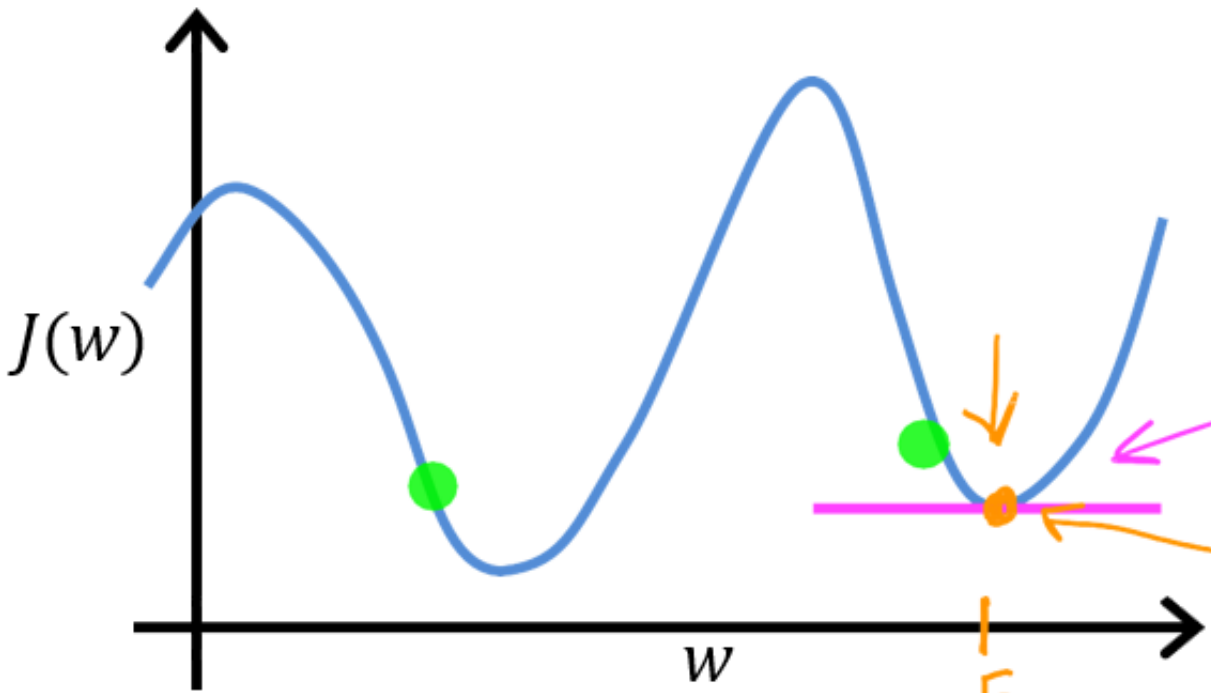
If $\alpha$ is too large...

Gradient descent may:

- Overshoot, never reach minimum
- Fail to converge, diverge






slope = 0

local minimum

current value of $w$ → 5

$$w = w - \alpha \frac{d}{dw} J(w)$$

0.1

$= 0$

$5 - 0.1 \cdot 0$

$w = w - \alpha \cdot 0$

$w = w$

# Can reach local minimum with fixed learning rate

$$w = w - \alpha \frac{d}{dw} J(w)$$

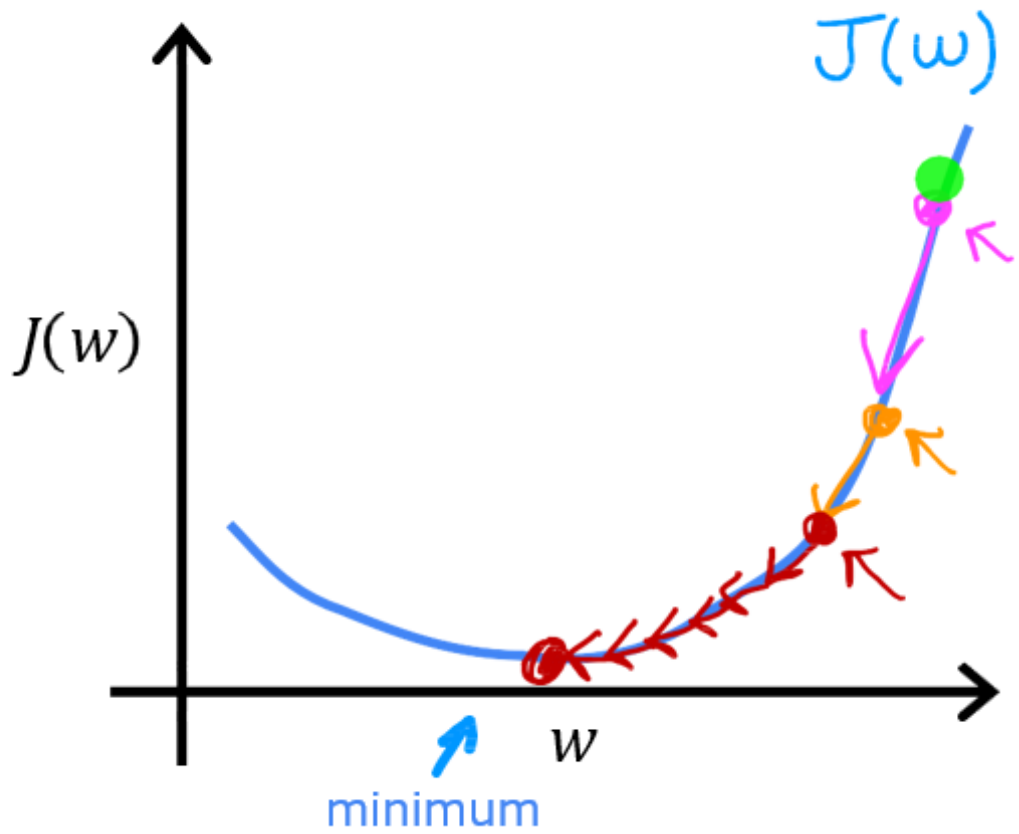
smaller

not as large

large

Near a local minimum,
- Derivative becomes smaller
- Update steps become smaller

Can reach minimum without decreasing learning rate $\alpha$



Linear regression model

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

Cost function

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

Gradient descent algorithm

repeat until [illegible] {

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

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

}

(Optional)

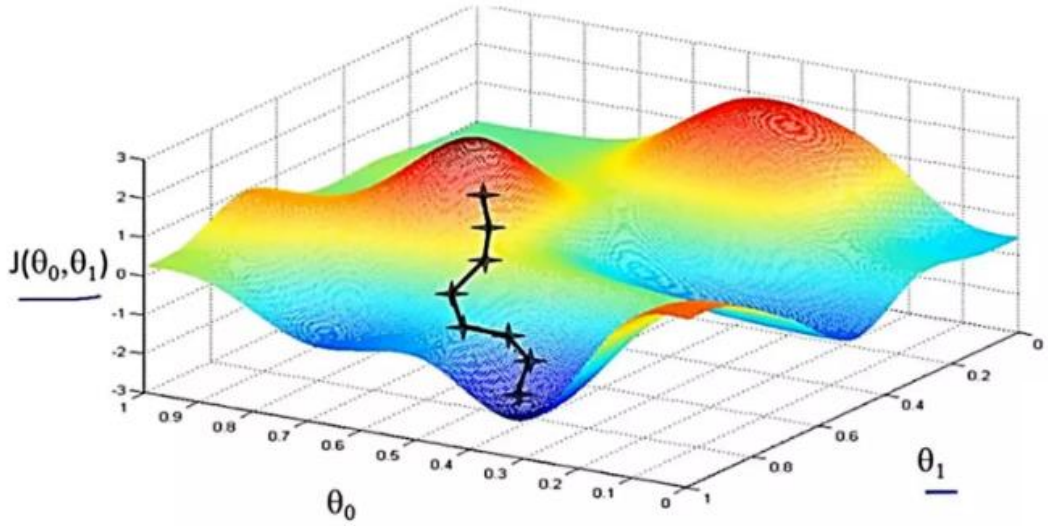
$$\frac{\partial}{\partial w} J(w,b) = \frac{d}{dw}\frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2 = \frac{d}{dw}\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)^2$$

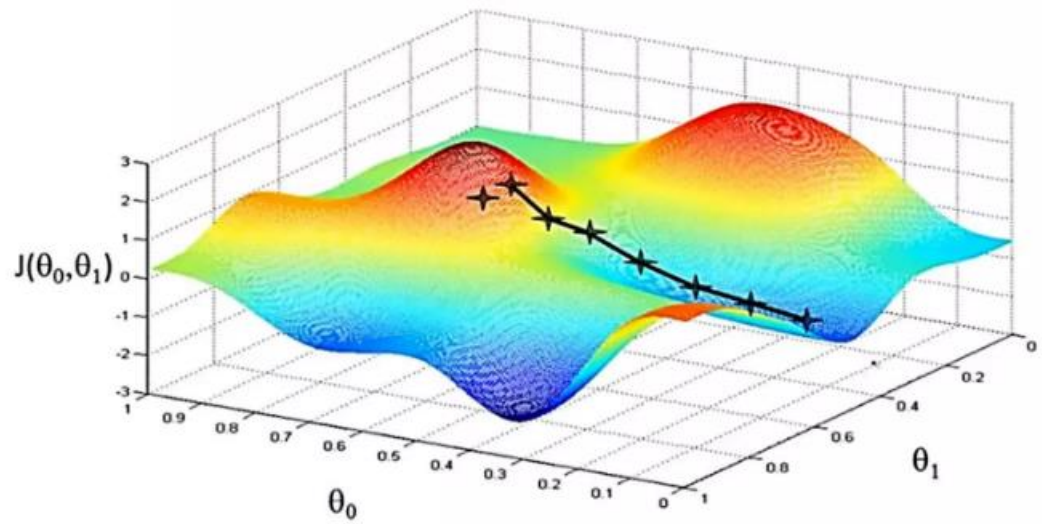
$$= \frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)2x^{(i)} = \boxed{\frac{1}{m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)x^{(i)}}$$

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

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

no $x^{(i)}$


J(θ0,θ1)
θ0
θ1


J(θ0,θ1)
θ0
θ1

# Gradient descent algorithm

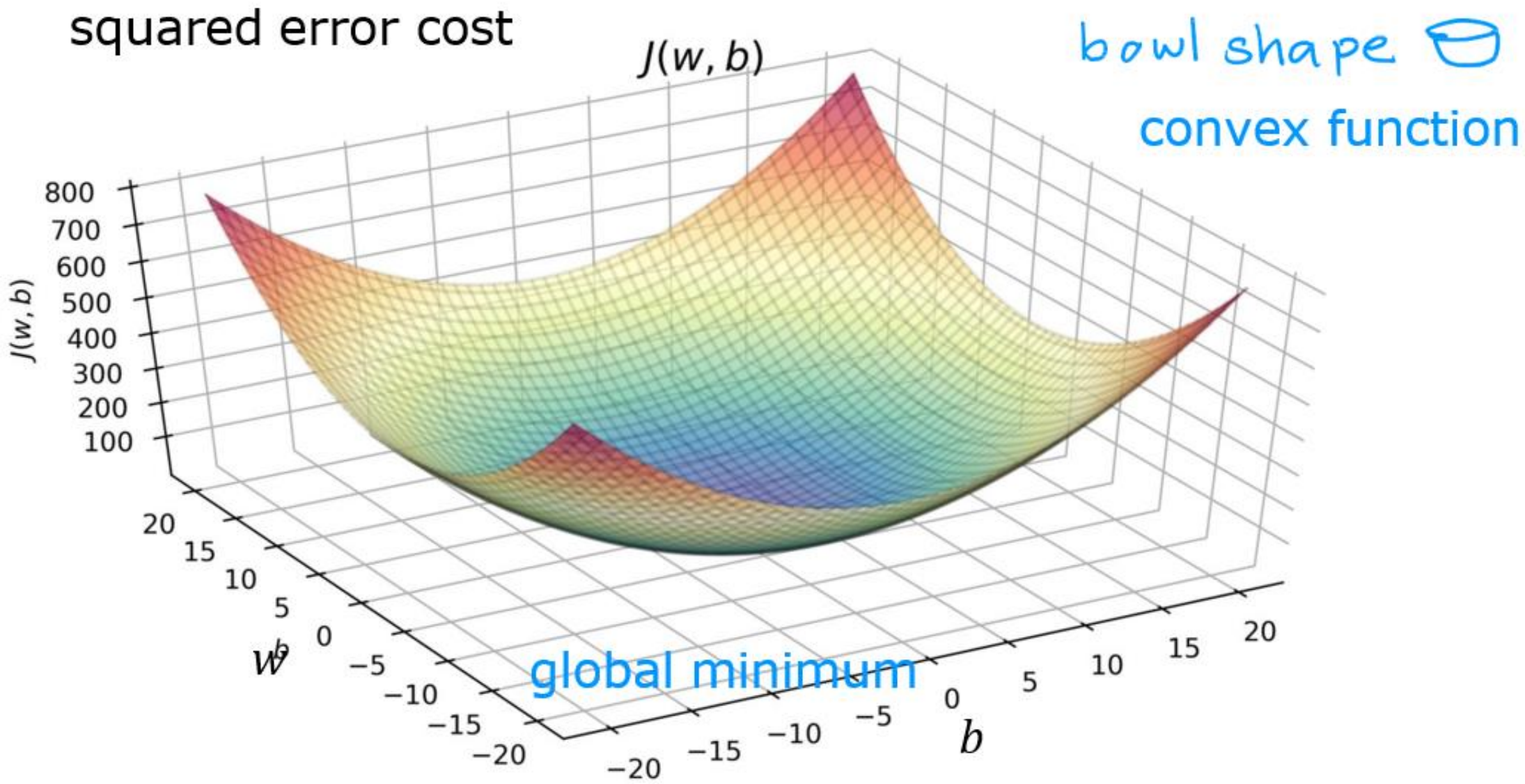
repeat until convergence {

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

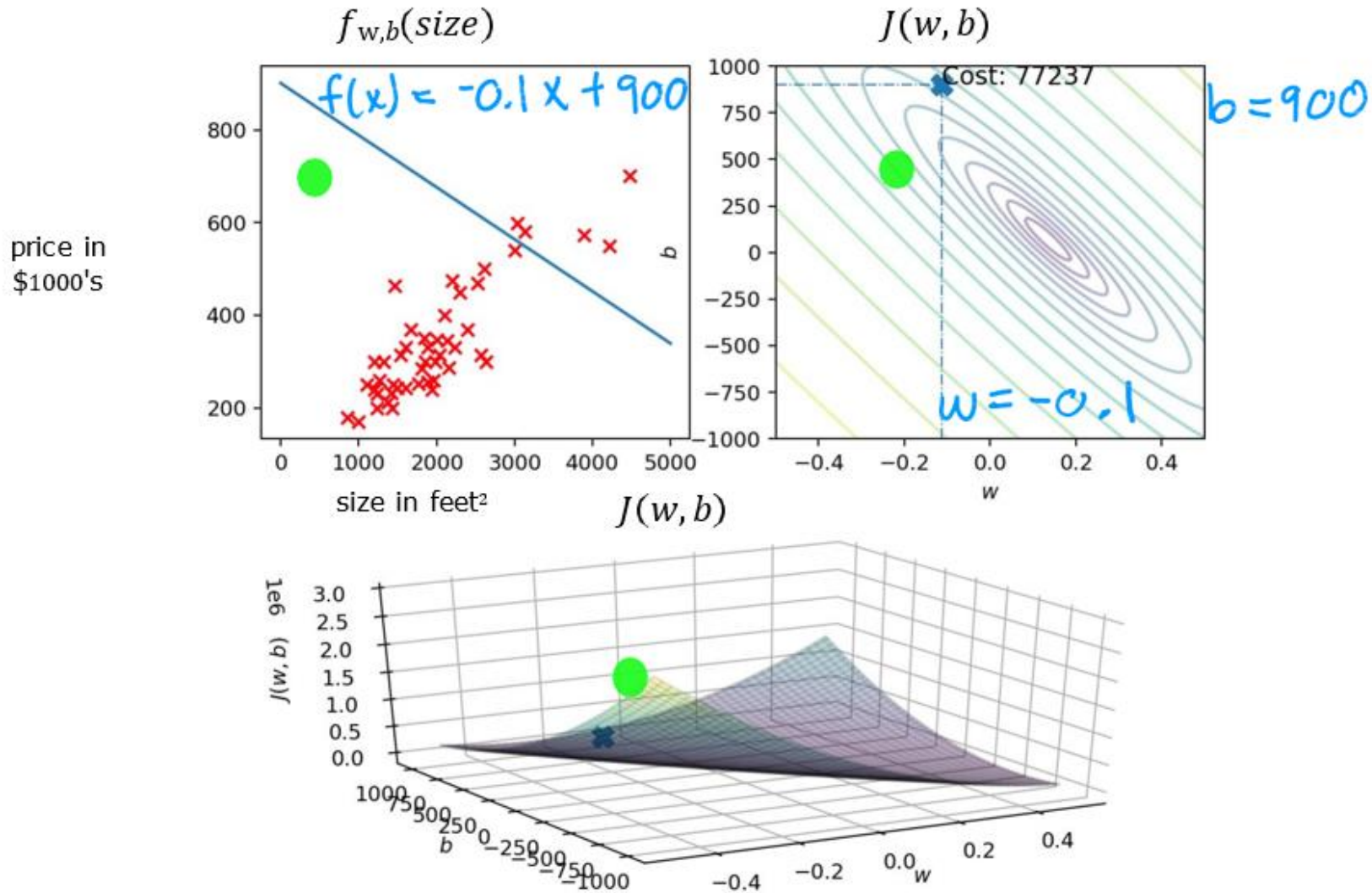
$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( f_{w,b}\left(x^{(i)}\right) - y^{(i)} \right)$$

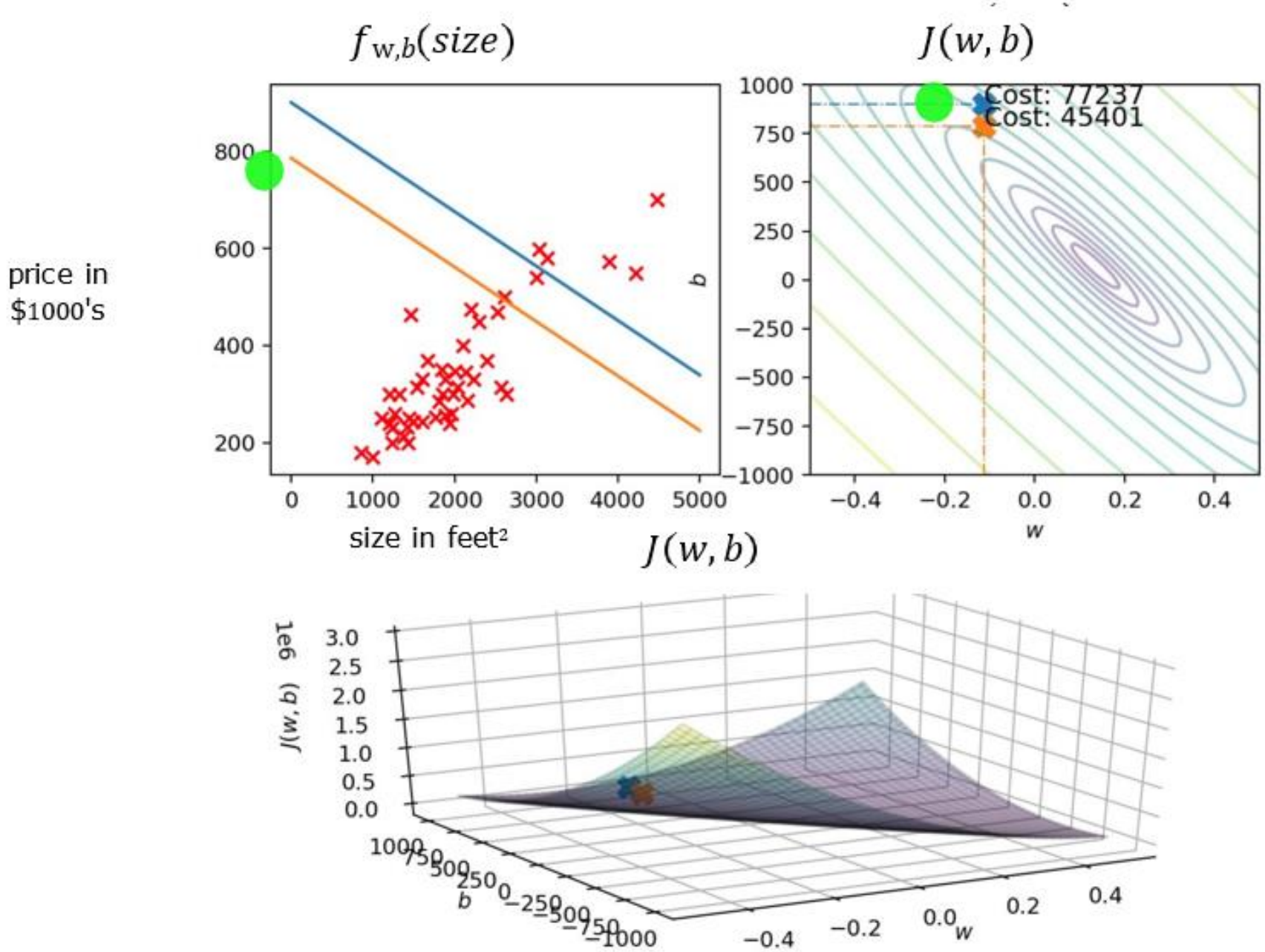
}

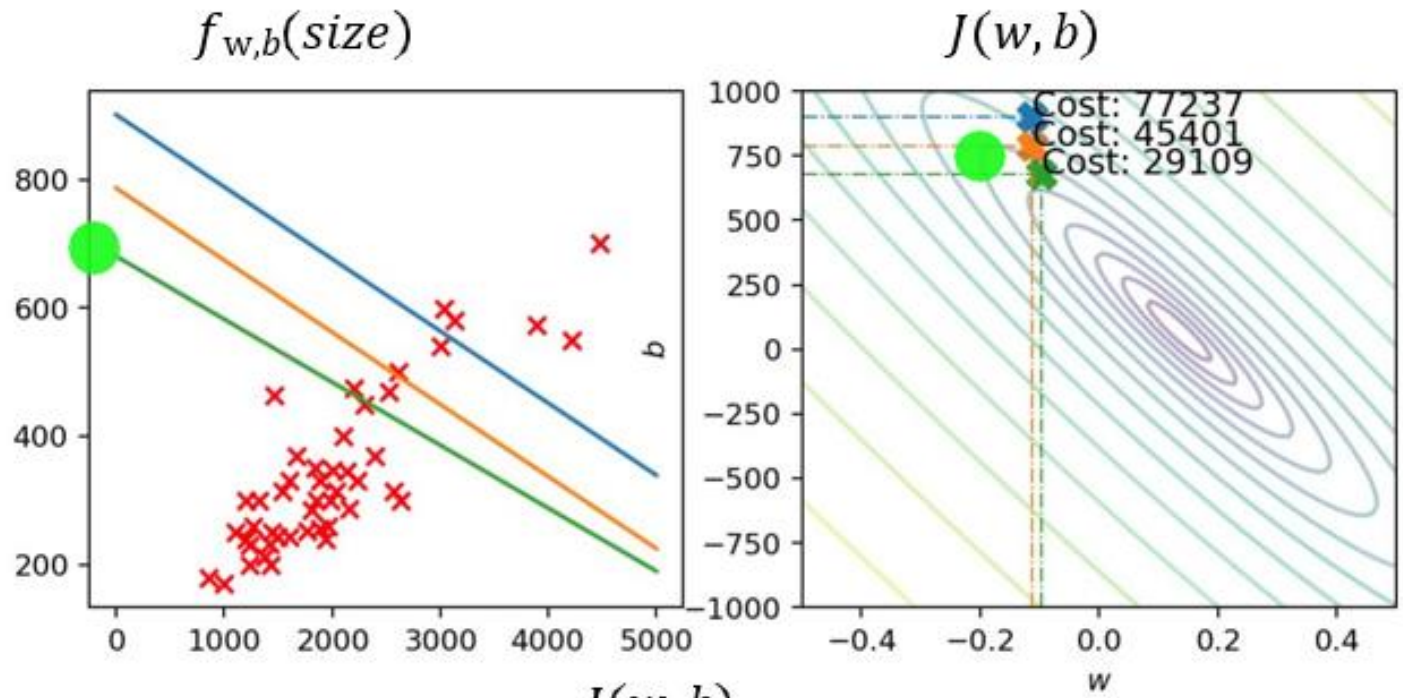
Update $w$ and $b$ simultaneously

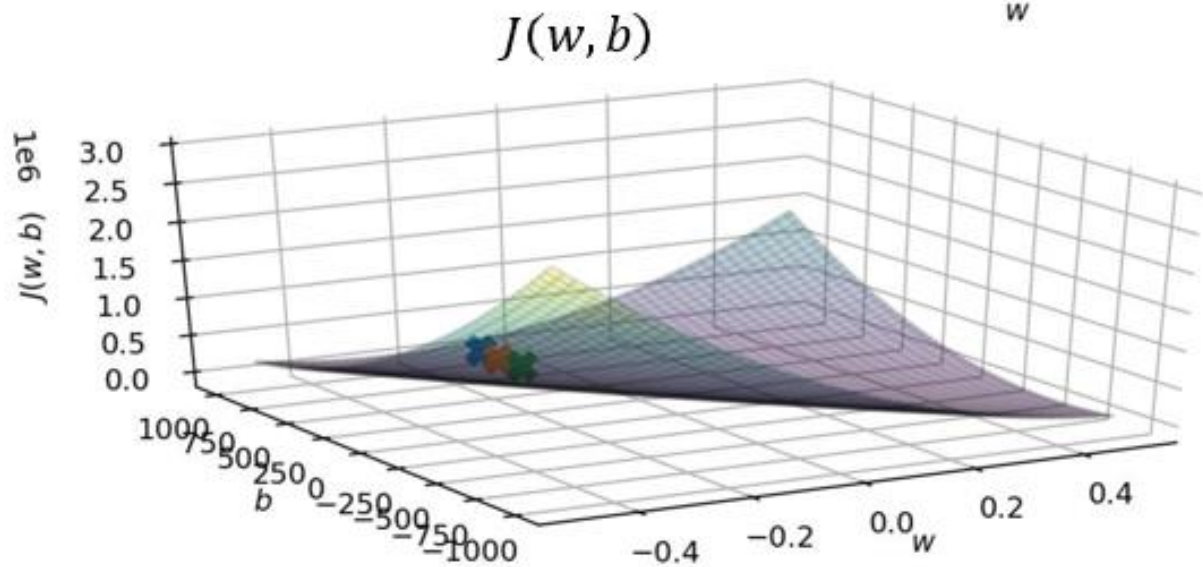

squared error cost
$J(w,b)$
bowl shape
convex function
800
700
600
500
400
300
200
100
$J(w,b)$
20
15
10
5
0
−5
−10
−15
−20
$w$
−20
−15
−10
−5
0
5
10
15
20
$b$
global minimum

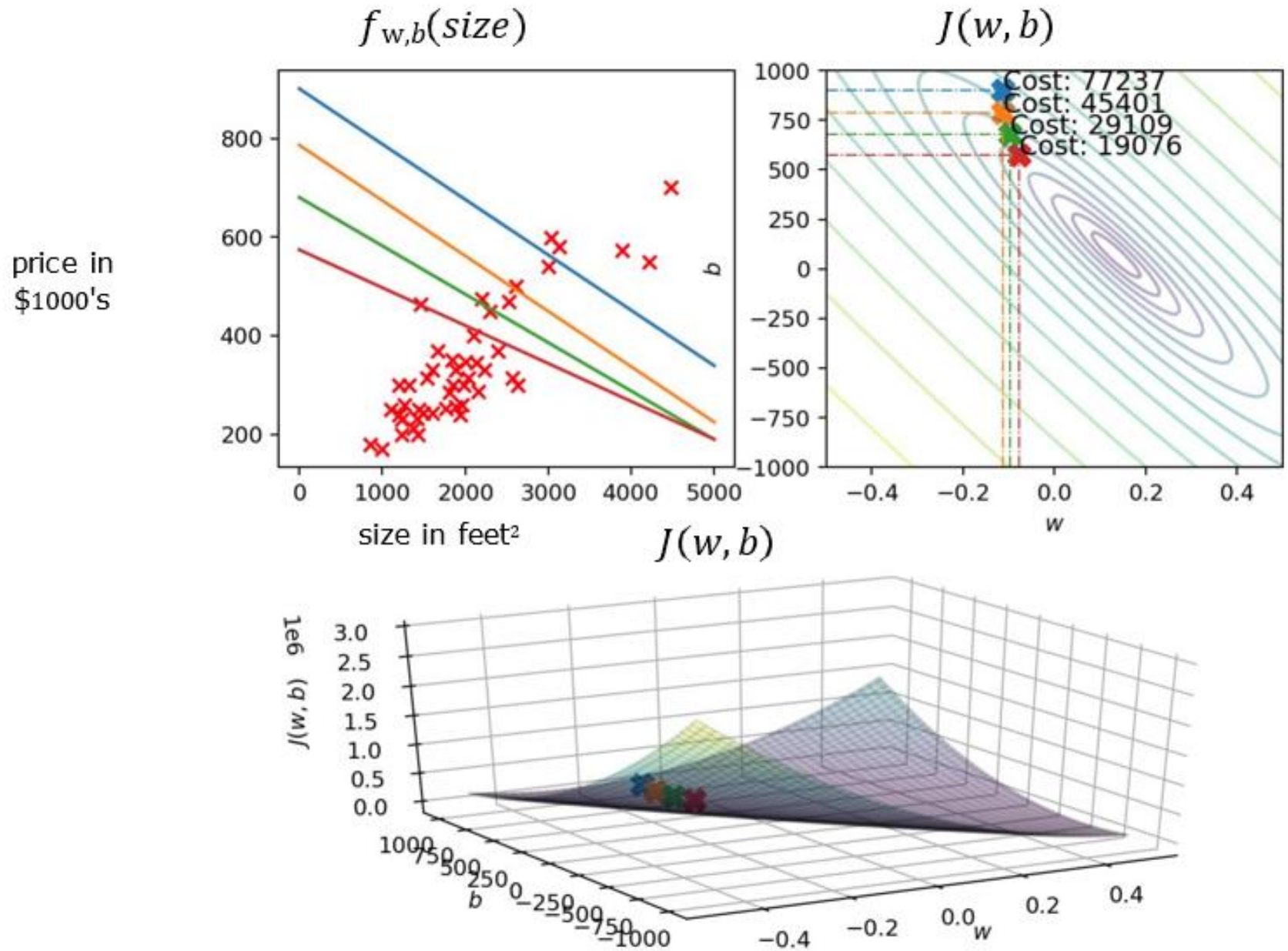
# Running Gradient Descent Algorithm

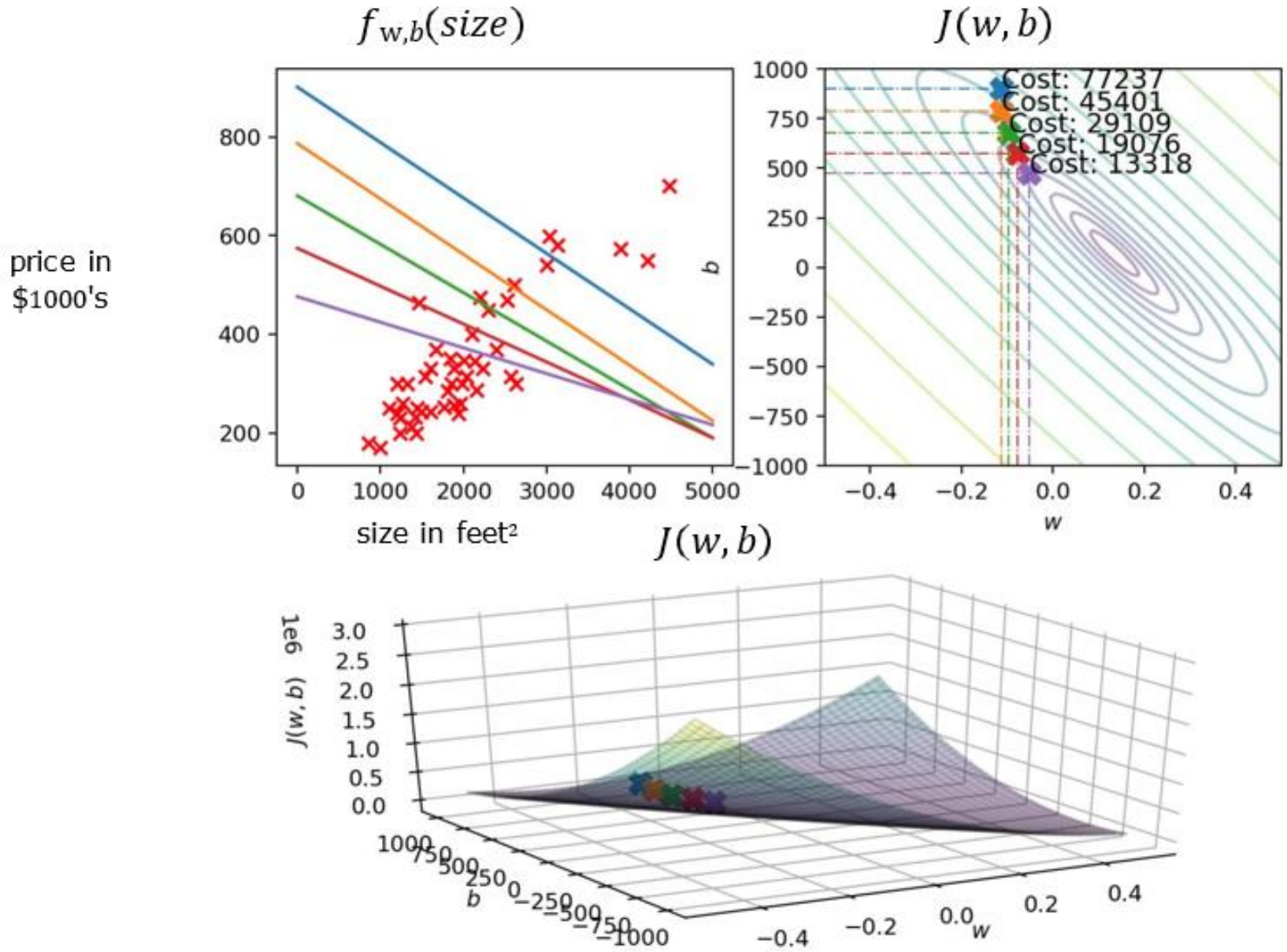


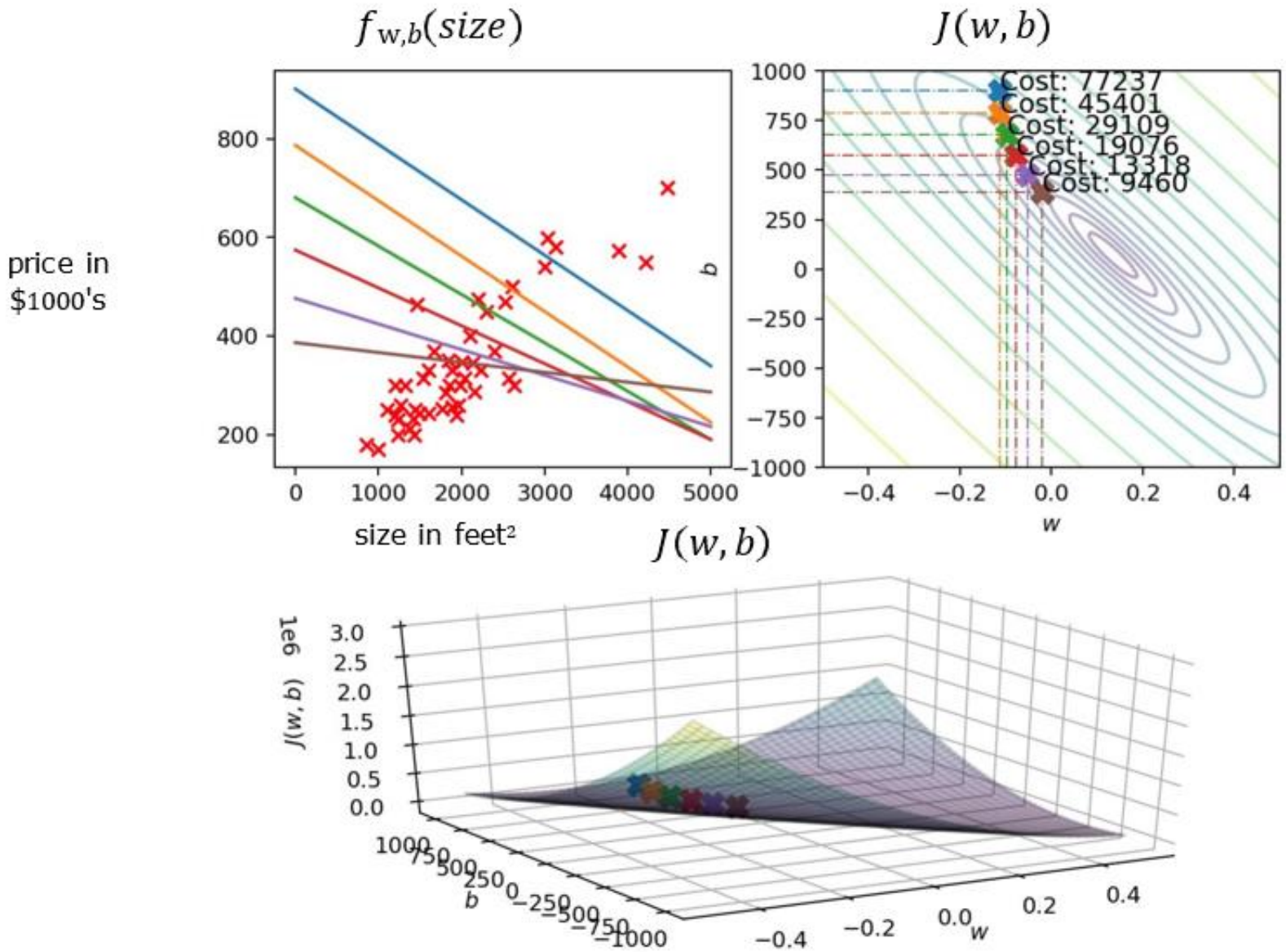

$f_{w,b}(size)$
$J(w,b)$
price in $1000's
size in feet²
Cost: 77237
Cost: 45401
b
w
$J(w,b)$
1e6
$J(w,b)$
b
w

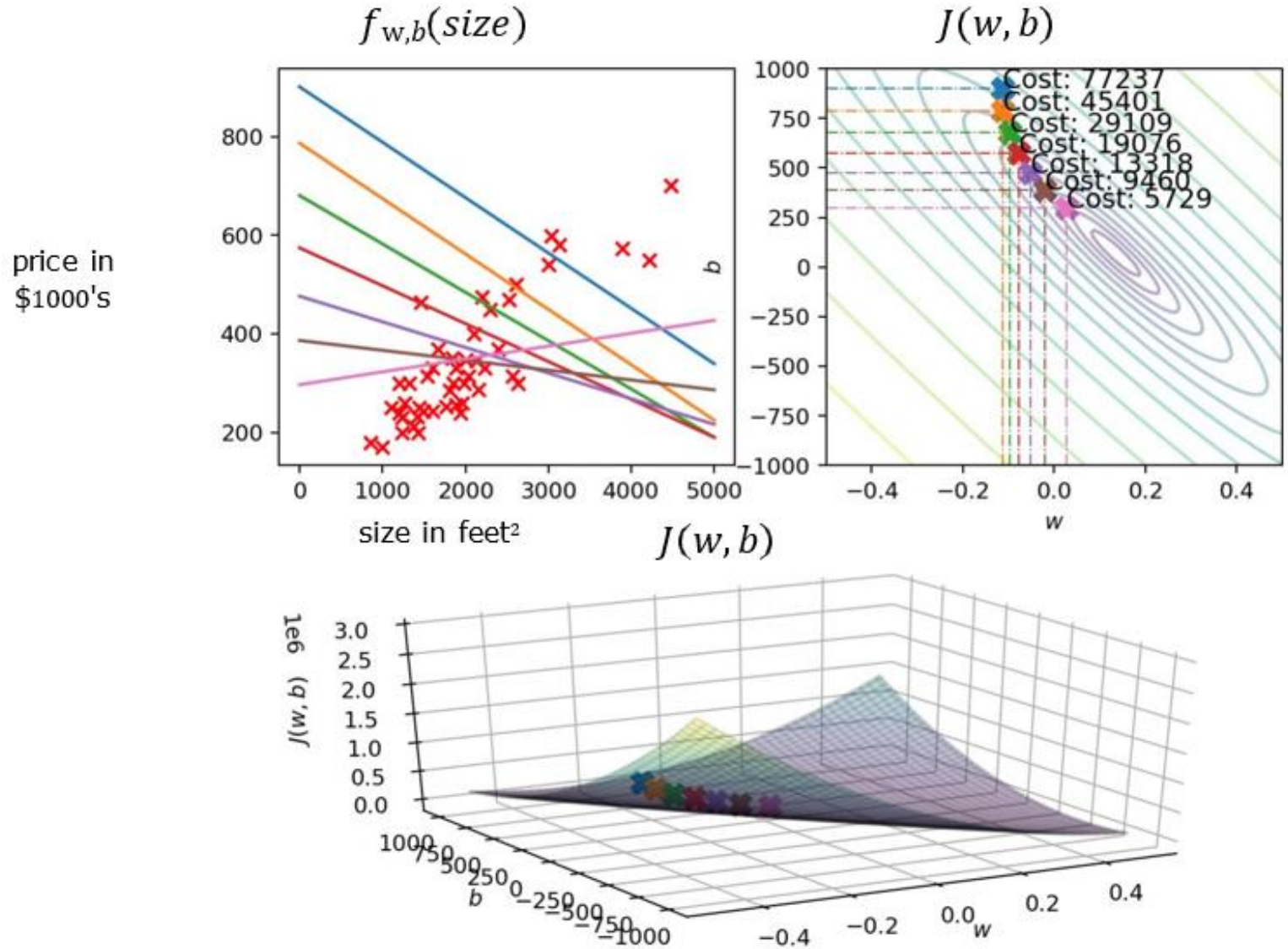

$f_{w,b}(size)$
$J(w,b)$
Cost: 77237
Cost: 45401
Cost: 29109
b
w

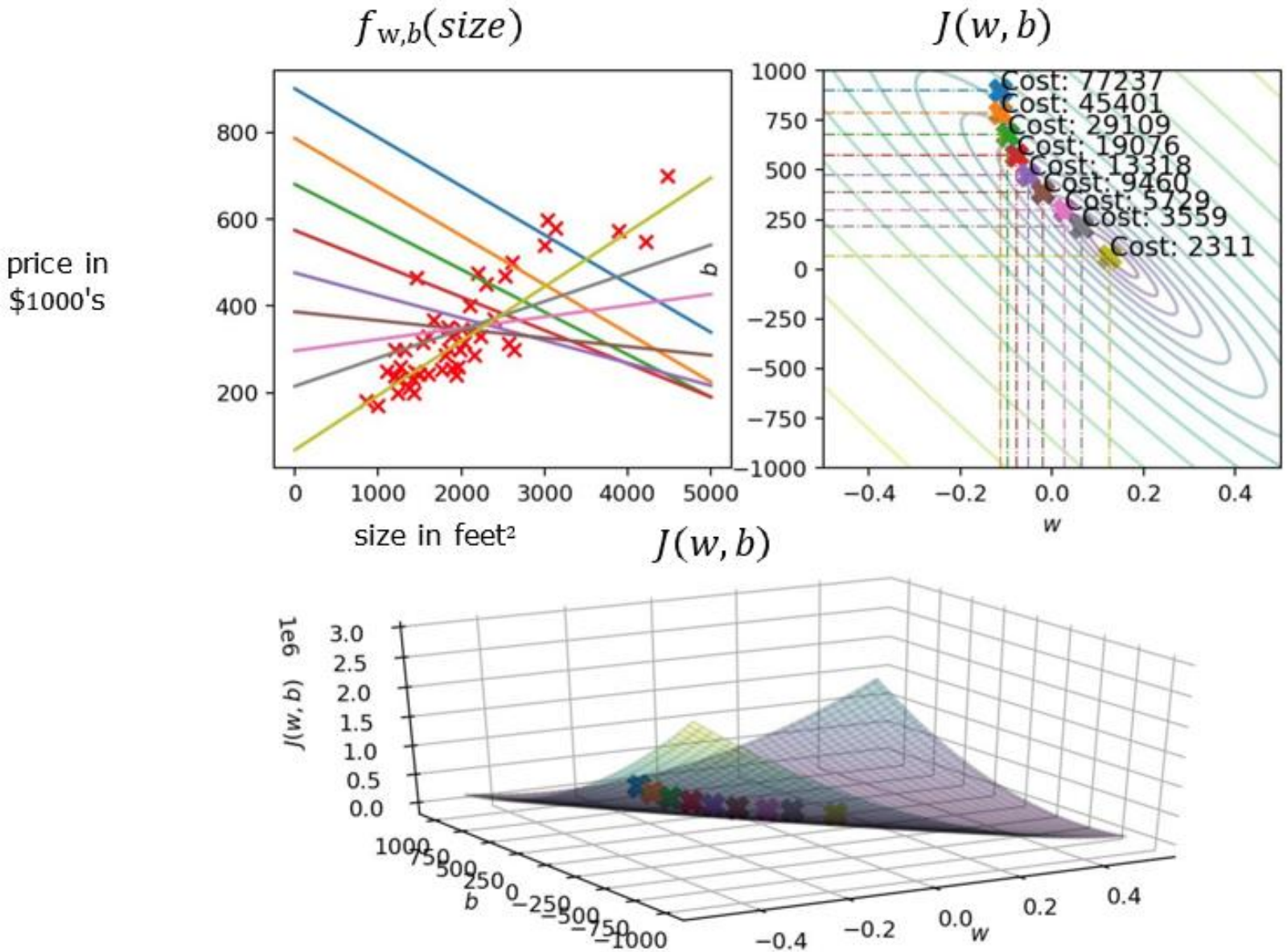

$J(w,b)$
1e6
$J(w,b)$
b
w


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


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


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


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


$f_{w,b}(size)$
price in $1000's
800
600
400
200
0
1000
2000
3000
4000
5000
size in feet²
b
$J(w,b)$
1000
750
500
250
0
−250
−500
−750
−1000
Cost: 77237
Cost: 45401
Cost: 29109
Cost: 19076
Cost: 13318
Cost: 9460
Cost: 5729
Cost: 3559
Cost: 2311
−0.4
−0.2
0.0
0.2
0.4
w
$J(w,b)$
1e6
3.0
2.5
2.0
1.5
1.0
0.5
0.0
$J(w,b)$
1000
750
500
250
0
−250
−500
−750
−1000
b
−0.4
−0.2
0.0
0.2
0.4
w