

COMP4434 Big Data Analytics

Lecture 3 Machine learning

HUANG Xiao

xiaohuang@comp.polyu.edu.hk

Definition of Machine Learning

- H. Simon

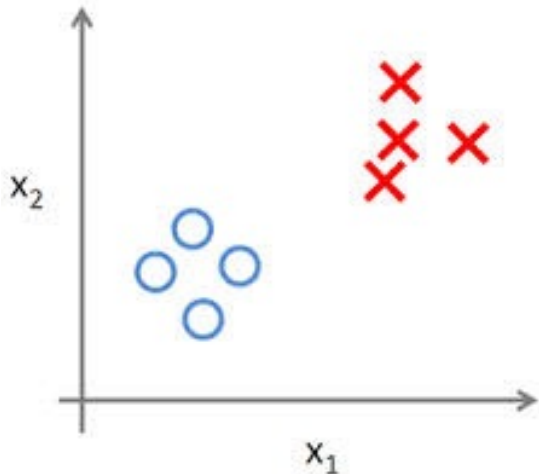
Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the task or tasks drawn from the same population more efficiently and more effectively the next time.

- T. Mitchell: Well posed machine learning –
Improving performance via experience

Formally, 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 .

Machine Learning Algorithms

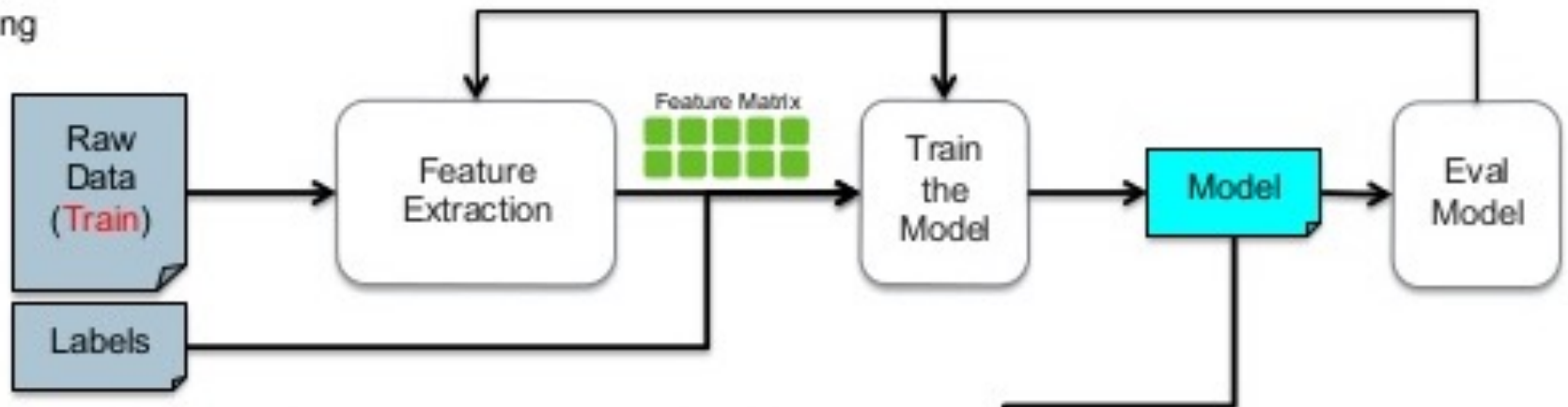
- Supervised Learning
 - Training data includes desired outputs



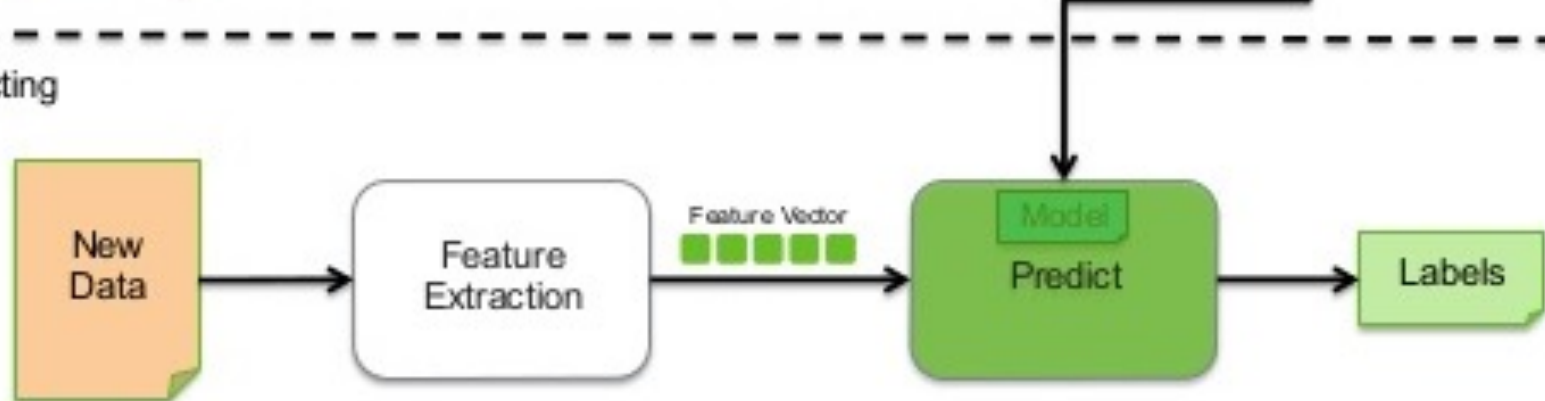
- Unsupervised Learning
 - Training data does not include desired outputs
 - Find hidden structure in data
- Semi-supervised Learning
- Reinforcement Learning

Supervised Learning Workflow

Training

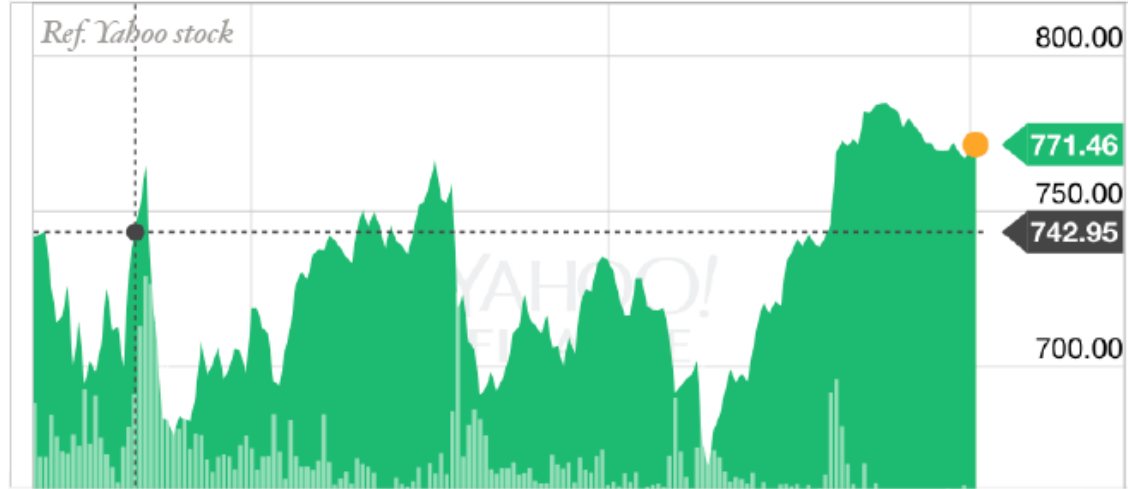


Predicting



Supervised Learning Task - Regression

- Regression
 - Learning to predict a continuous/real value
 - Ex: housing price, gold price, stock price



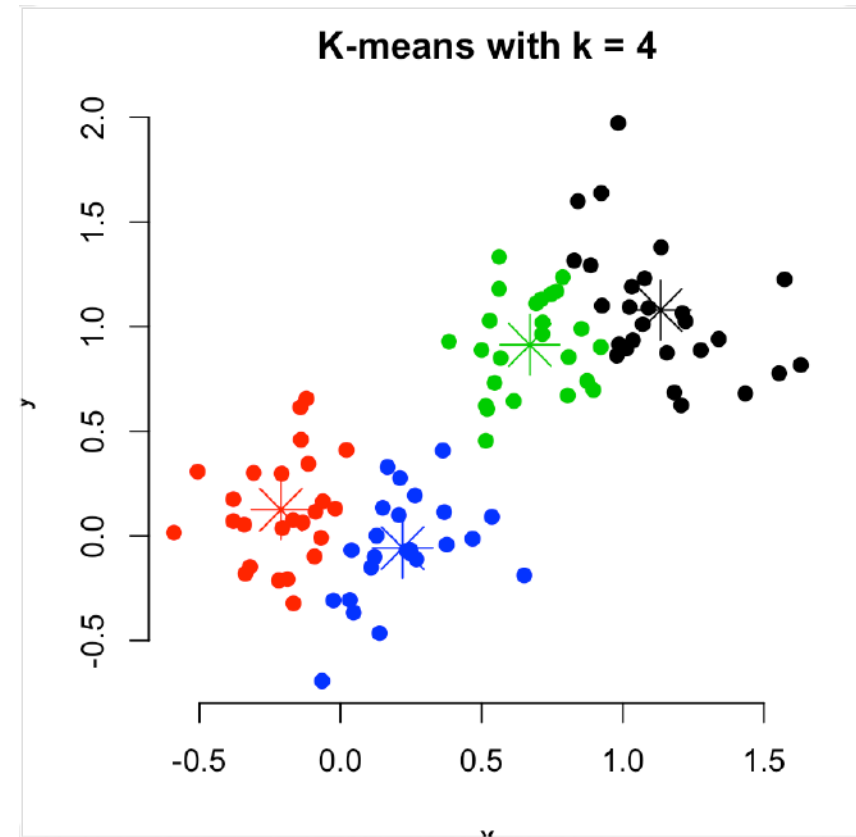
Supervised Learning Task - Classification

- Classification
 - Learning to predict a discrete value from a predefined set of values
 - Ex. weather prediction, spam email filtering, product categorization, object detection, medical diagnose



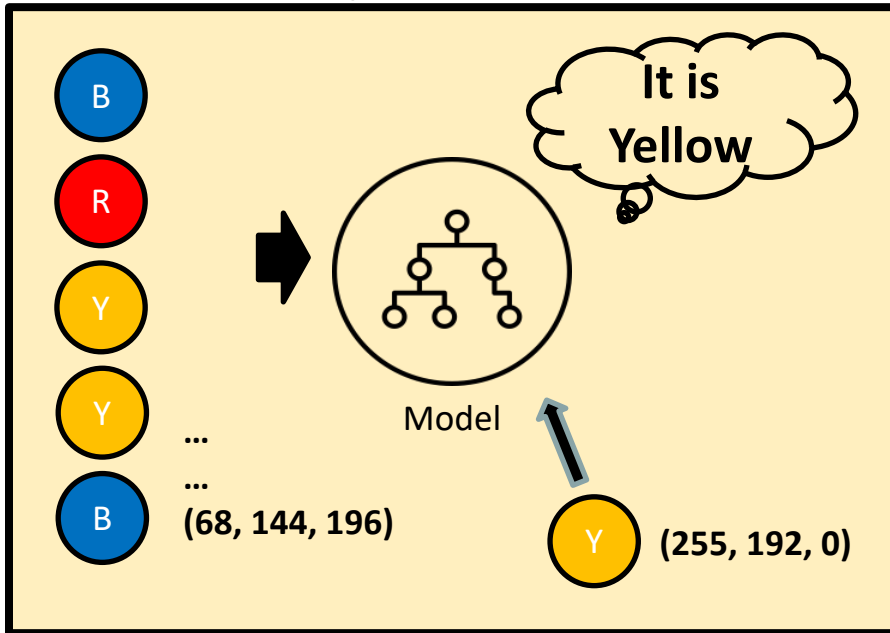
Unsupervised Learning Task - Clustering

- Clustering
 - Determine the intrinsic grouping in a set of unlabeled data
 - Ex. clustering in networking, image clustering



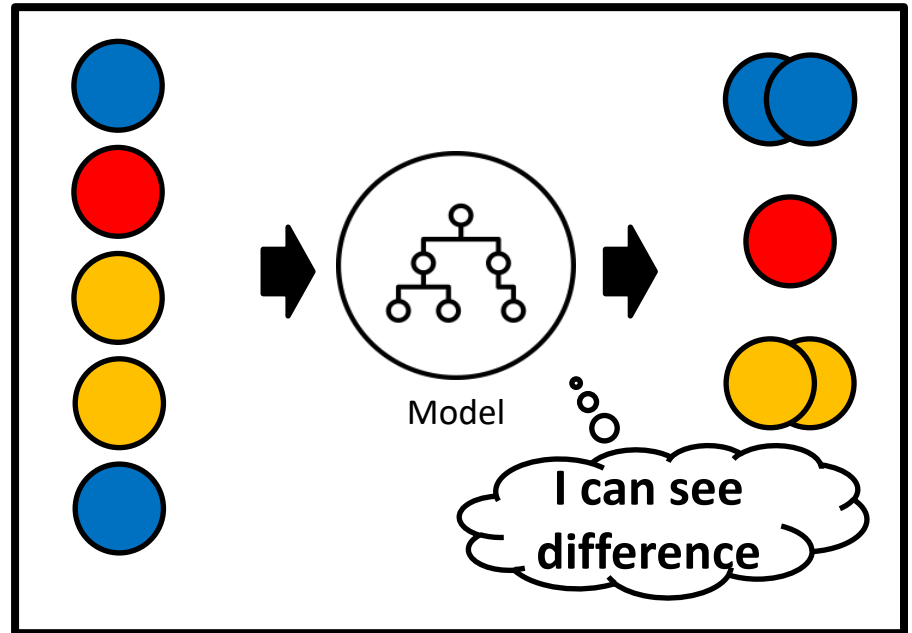
Supervised vs Unsupervised

Supervised



- Labeled Data
- Direct Feedback
- Predict Output

Unsupervised

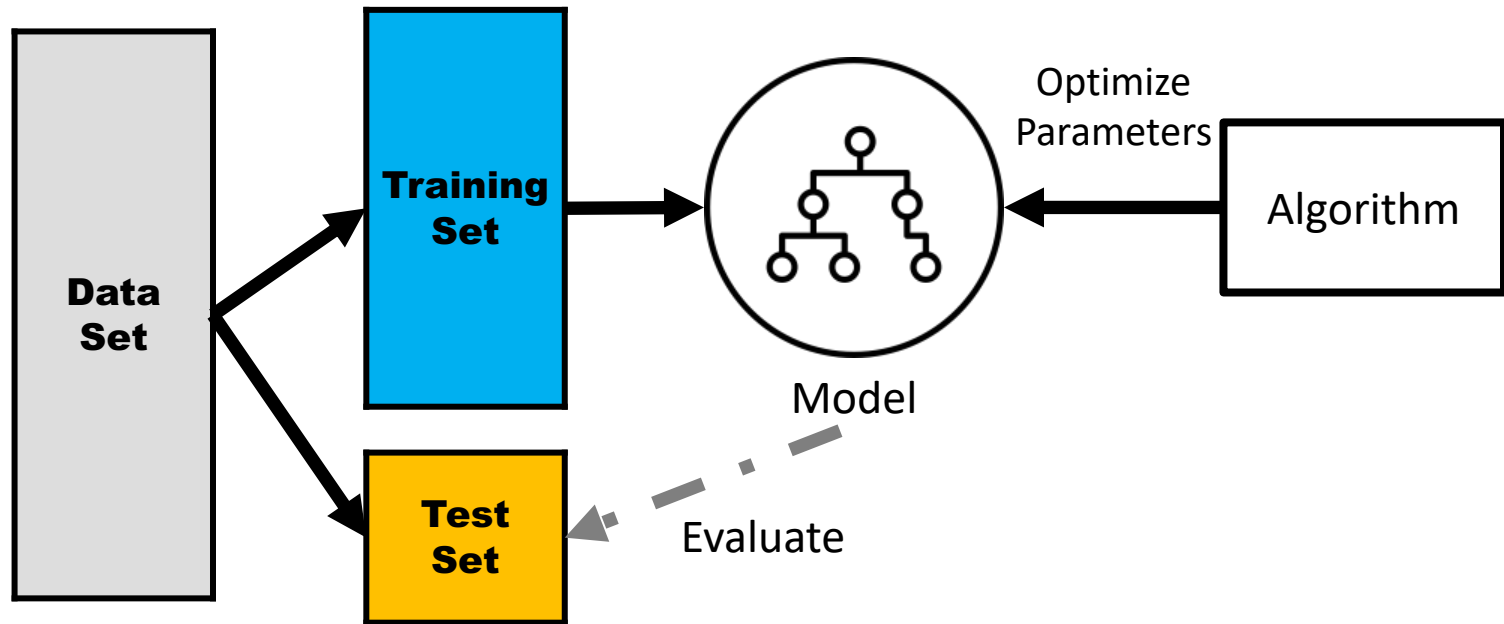


- Non-labeled Data
- No Feedback
- Find Hidden Structure in Data

Supervised learning tasks

- Classification
 - predicts **categorical** class labels
 - classifies data (constructs a model) based on the training set and the values (class labels) and uses the trained model to classify new data
 - return a discrete-value (label) as output, e.g., classifying Hang Seng Index (HSI)'s trend as Up, Down, Level
- Regression
 - models **continuous-valued** functions, i.e., predicts unknown or missing values
 - Return a real-value as output, e.g., predicting HSI's future values

Two-Step Process



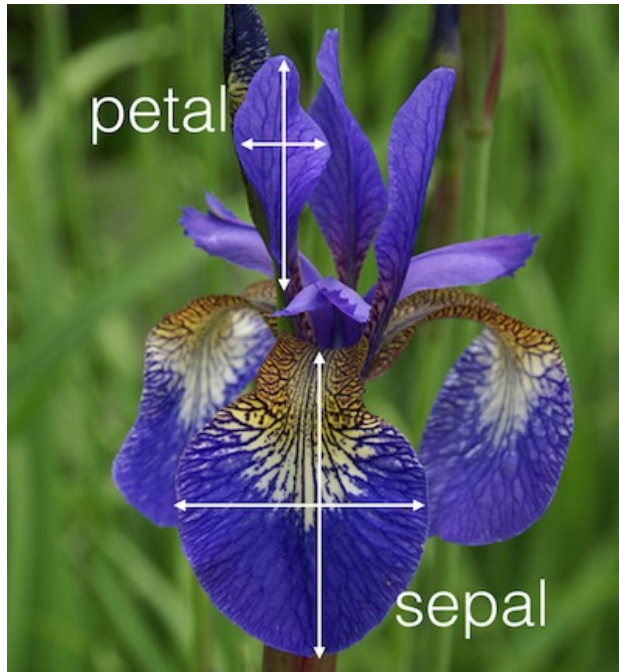
1. Model Construction: describing a set of samples

- Each sample is associate with a label attribute
- The set of samples used for model construction: **training set**
- The model is represented as mathematical formulae

2. Model Usage: for future or unknown objects

- The known label of test sample is compared with the result from the model
- Test set is **independent** of training set

Structured Data: Tabular Data



Sepal		Petal	
Length	Width	Length	Width
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.8	4.0	1.2	0.2
5.6	3.0	4.5	1.5
5.8	2.7	4.1	1.0
6.2	2.2	4.5	1.5
5.6	2.5	3.9	1.1
5.9	3.2	4.8	1.8
7.1	3.0	5.9	2.1
6.3	2.9	5.6	1.8
6.5	3.0	5.8	2.2
7.6	3.0	6.6	2.1

Instance/tuple

attribute/feature

Tabular Data with Labels

- Attributes of a table
 A_1, A_2, \dots, A_d
 - We also call “**attribute**” as “**feature**”
 - Number of features
 d represents the dimensionality
 - A data object x is represented as
 (x_1, x_2, \dots, x_d)
- In some datasets, each data object has a **label**

Sepal		Petal		Class Label
Length	Width	Length	Width	
4.9	3.0	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
4.6	3.1	1.5	0.2	Setosa
5.0	3.6	1.4	0.2	Setosa
5.8	4.0	1.2	0.2	Setosa
5.6	3.0	4.5	1.5	Versicolor
5.8	2.7	4.1	1.0	Versicolor
6.2	2.2	4.5	1.5	Versicolor
5.6	2.5	3.9	1.1	Versicolor
5.9	3.2	4.8	1.8	Versicolor
7.1	3.0	5.9	2.1	Virginica
6.3	2.9	5.6	1.8	Virginica
6.5	3.0	5.8	2.2	Virginica
7.6	3.0	6.6	2.1	Virginica
4.9	2.5	4.5	1.7	Virginica



Iris Versicolor



Iris Setosa



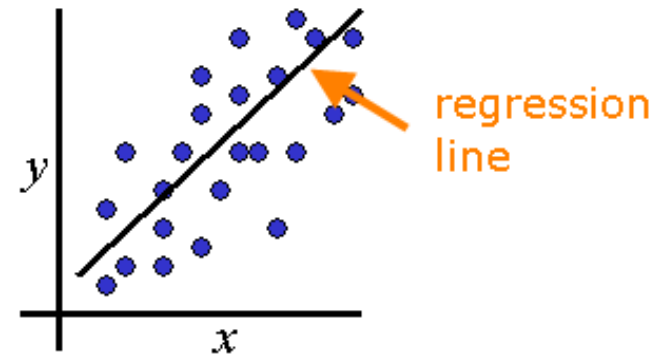
Iris Virginica

Classification (Binary vs Multi-class)

- Binary Classification
 - Email: Spam / Not Spam
 - Tumor: Malignant / Benign
 - Covid-19: Positive / Negative
 - $y \in \{0,1\}$
- Multi-class Classification
 - Email auto-tagging: Spam / Work / Personal
 - Credit Rating: Poor / Okay / Trust
 - Handwriting number: 0, 1, 2, 3, 4, ...
 - $y \in \{0,1,2, \dots\}$

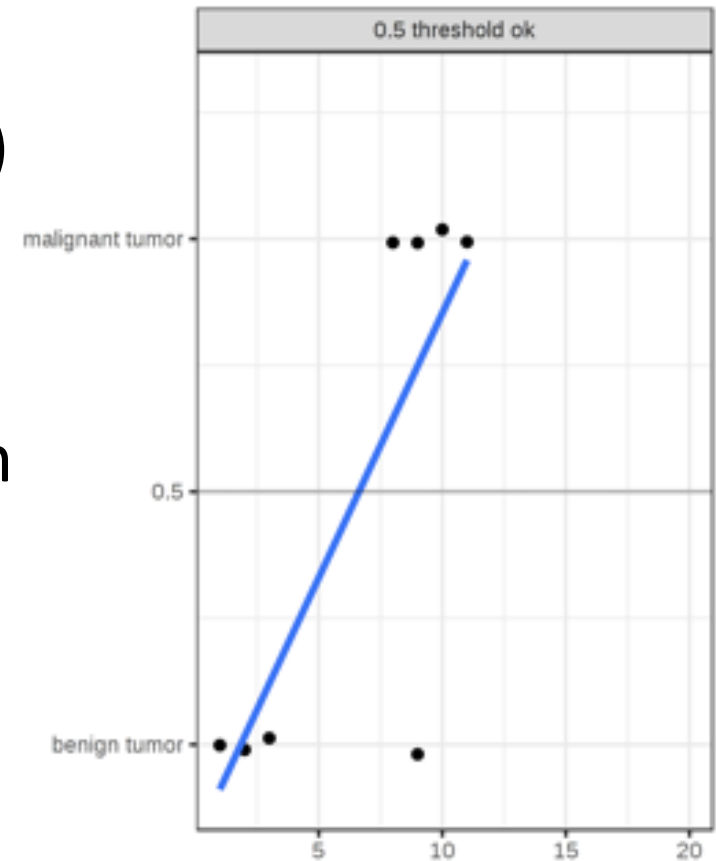
Classification by Linear Regression

- Training Examples
 - both are supervised learning (x, y)
 - Linear Regression: y is a real-value, e.g., salary
- What we need is discrete label:
 - 0: malignant; 1: benign
- Can we use Linear Regression Model to do classification? Any disadvantages?
 - Yes, we can, but not good
 - $h_{\theta}(x) = \theta_0 + \theta_1 x$



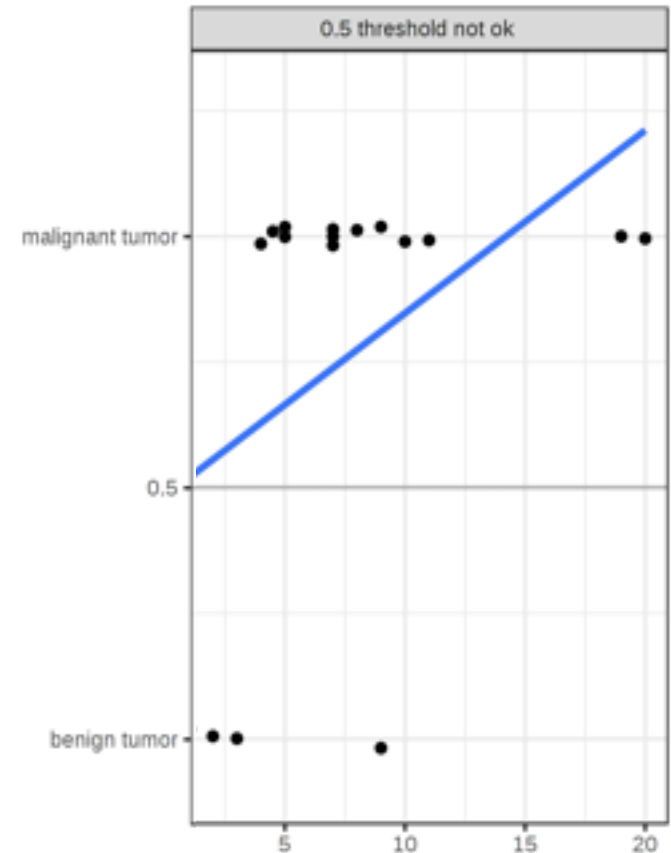
Classification by Linear Regression - Example

- A linear regression model classifies tumors as malignant (1) or benign(0) given their size
- The linear regression model minimizes the distances between the points and the hyperplane (line for single feature)
- The threshold is set as 0.5



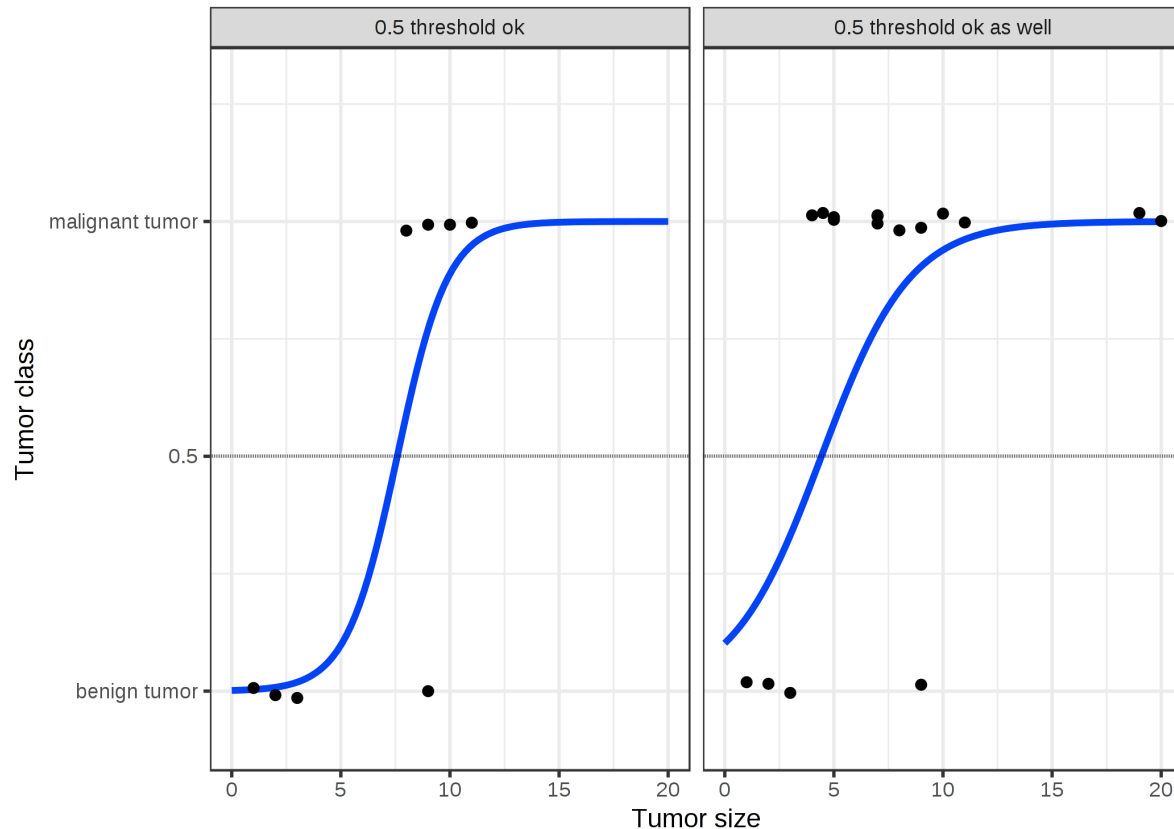
Classification by Linear Regression - Example

- After introducing a few more malignant tumor cases, the regression line shifts and a threshold of 0.5 no longer separates the classes
- **Conclusion: Linear regression is sensitive to imbalance data for classification problem.**



Logistic Regression

- New model outputs **probabilities**
- It works better in both cases using 0.5 as a threshold
- The inclusion of additional points does not affect the estimated curve too much



Hypothesis Model

- Linear Regression: $-\infty < h_{\theta}(x) < +\infty$

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \dots + \theta_n x_n = \theta^T x$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \quad x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

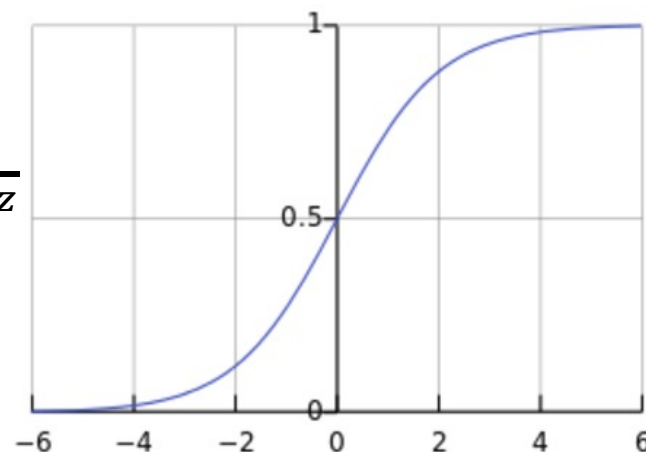
- Logistic Regression: $0 < h_{\theta}(x) < 1$

- $h_{\theta}(x) = g(\theta^T x)$

- logistic/sigmoid function** $g(z) = \frac{1}{1+e^{-z}}$

- $h_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}}$

- [https://www.wolframalpha.com/](https://www.wolframalpha.com/y=1/(1+e^-x),x%20from%20-6%20to%206)
 $y = 1/(1+e^{-x})$, x from -6 to 6

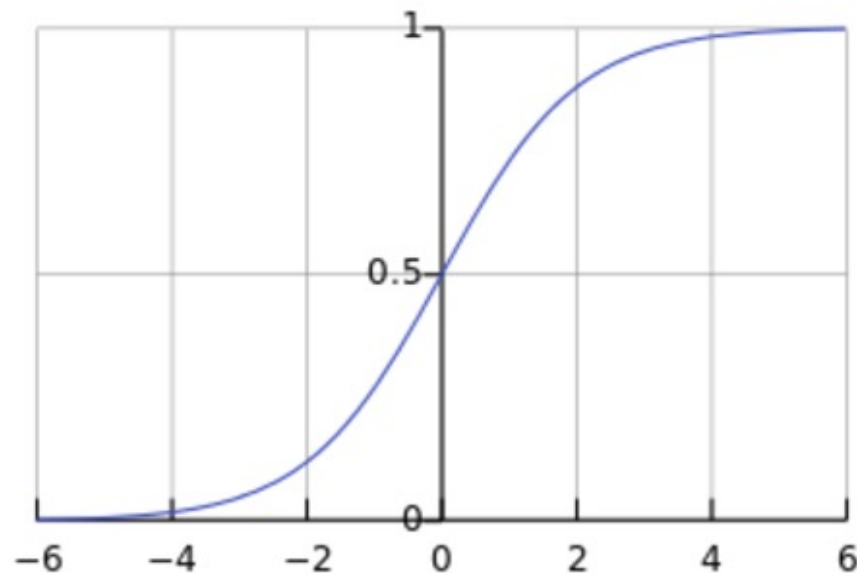


Representation

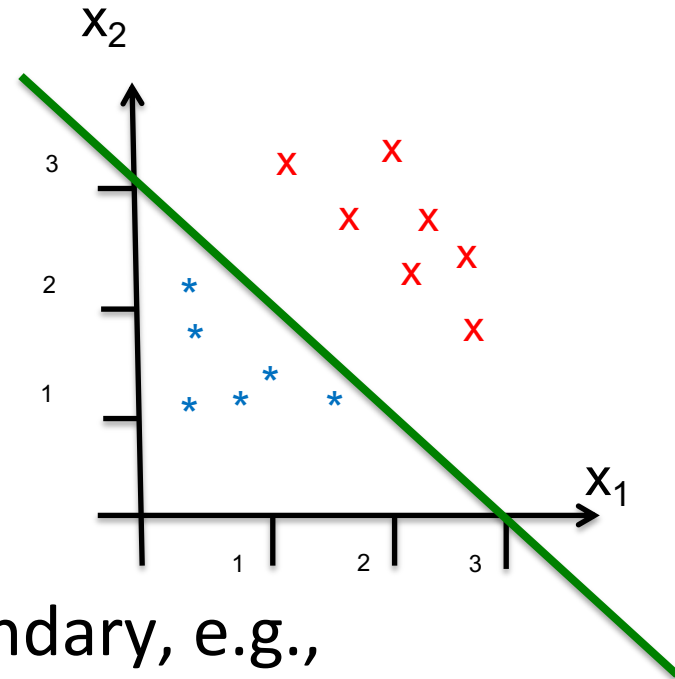
- $h_{\theta}(x)$ represents the estimated **probability** that $y = 1$ on input x
- $h_{\theta}(x) = P\{y = 1|x, \theta\}$ means probability of $y = 1$, given x , under parameter values θ
- $P\{y = 0|x, \theta\} = 1 - h_{\theta}(x)$
- Example
 - $\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{number of spam words} \end{bmatrix}$
 - $h_{\theta}(x) = 0.8$: this email x has 80% chance of being spam

Further Understanding

- $h_{\theta}(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}} \in (0,1)$
- Predict $y = 1$ when $h_{\theta}(x) \geq 0.5$, i.e., $\theta^T x \geq 0$
- Predict $y = 0$ when $\theta^T x < 0$



Decision Boundary



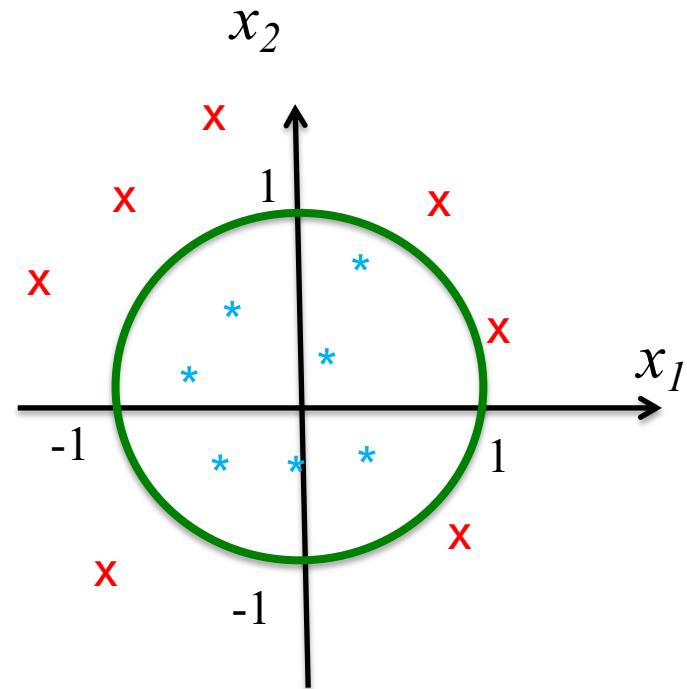
- $\theta^T x = 0$ is the decision boundary, e.g.,
 - Assume $h_\theta(x) = g(-3 + x_1 + x_2)$
 - Decision boundary: $-3 + x_1 + x_2 = 0$,
i.e., $x_1 + x_2 = 3$
 - Predict $y = 1$ when $-3 + x_1 + x_2 \geq 0$,
i.e., $x_1 + x_2 \geq 3$ (red)

Other Decision Boundary

- Given $h_{\theta}(x) = g(\theta^T x) = g(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2) = g(-1 + x_1^2 + x_2^2)$
- If $h_{\theta}(x) = g(-1 + x_1^2 + x_2^2)$, draw the region that predicts $y = 1$ in the (x_1, x_2) plane

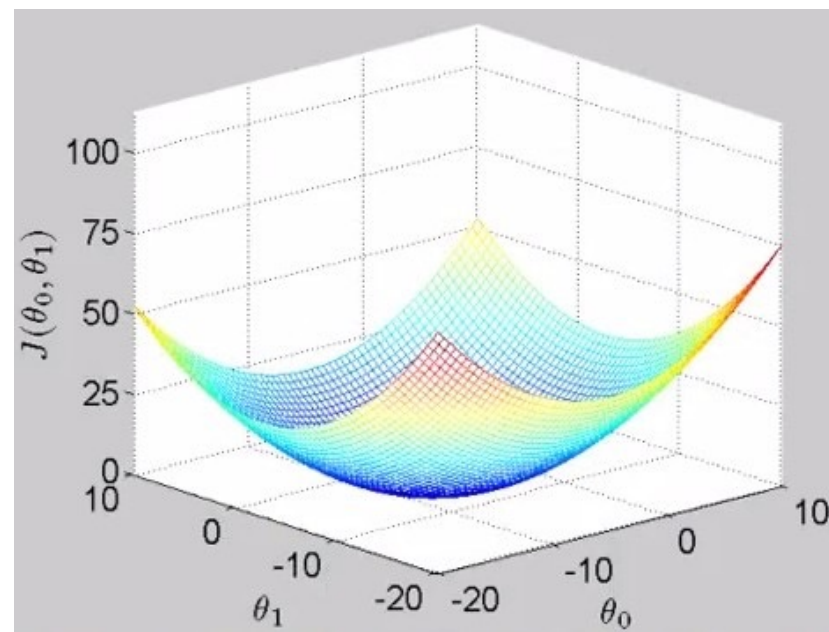
Non-Linear Decision Boundary

- Decision boundary $\theta^T x = 0 \rightarrow x_1^2 + x_2^2 = 1$
- Predict $y = 1$ when $x_1^2 + x_2^2 \geq 1$
- The region of $y = 1$ is outside the circle



Cost Function of Linear Regression

- Cost Function $J(\theta) = \min_{\theta_0, \theta_1, \dots} \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$
- Linear Regression
 - h_{θ} is **linear**
 - $J(\theta)$ is **convex**
 - $J(\theta)$ has a **single minimum**



Regression Metrics

- MSE: Mean Square Error

$$\frac{1}{n} \sum_{i=1}^n (\hat{y}_t - y_t)^2$$

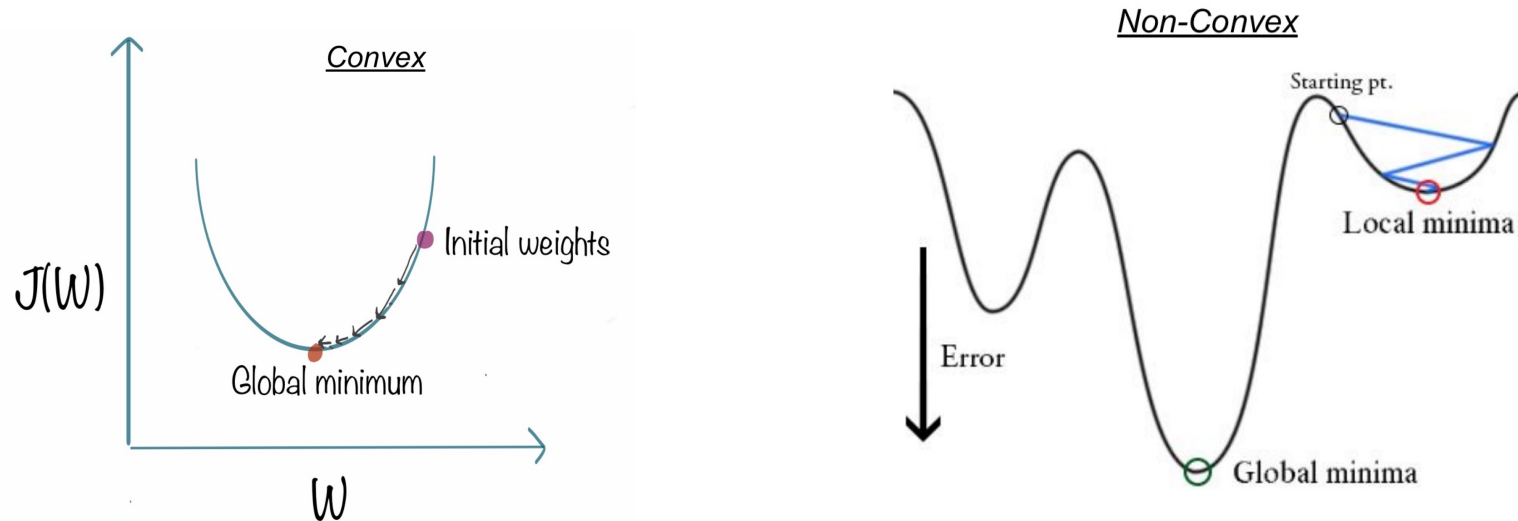
- MAE: Mean Absolute Error

$$\frac{1}{n} \sum_{i=1}^n |\hat{y}_t - y_t|$$

- MAPE (Mean Absolute Percentage Error)

$$\frac{100\%}{n} \sum_{i=1}^n \left| \frac{\hat{y}_t - y_t}{y_t} \right|$$

Apply MSE to Logistic Regression



- We can apply the same cost function for logistic regression
- Problems
 - $J(\theta)$ would become non-convex. Why?
 - It has multiple local minimums
 - Gradient descent will be stuck in a local minimum

<https://towardsdatascience.com/why-not-mse-as-a-loss-function-for-logistic-regression-589816b5e03c>

Logistic Loss

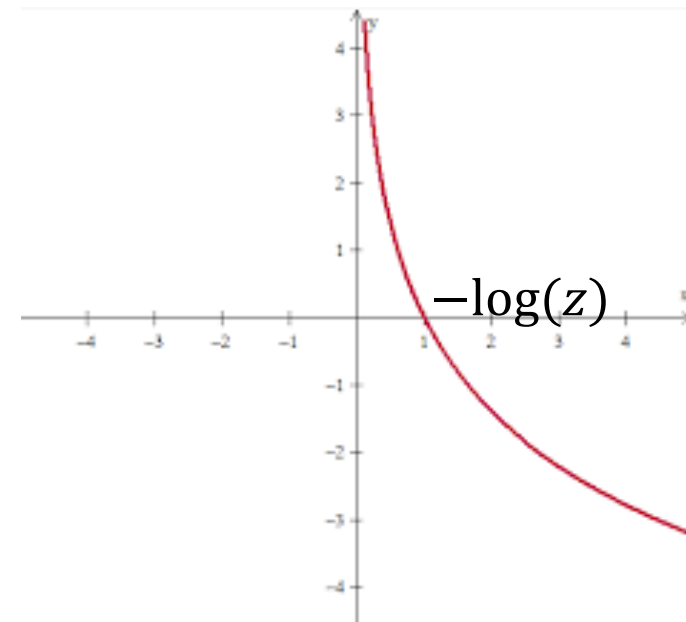
$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & y = 1 \\ -\log(1 - h_{\theta}(x)) & y = 0 \end{cases}$$

Logistic Loss - Heavy Penalty

$$Cost(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & y = 1 \\ -\log(1 - h_{\theta}(x)) & y = 0 \end{cases}$$

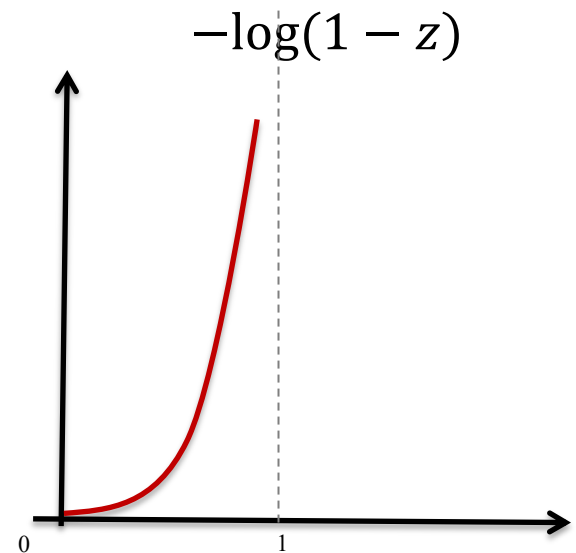
- $Cost(h_{\theta}(x), y) \rightarrow 0$
 - When $h_{\theta}(x) \rightarrow 1$, i.e., predict $y = 1$
 - Good predication, low cost
- $Cost(h_{\theta}(x), y) \rightarrow \infty$
 - When $h_{\theta}(x) \rightarrow 0$, i.e., predict $y = 0$
 - Bad predication!
 - High cost represents penalty



Understanding $Cost(h_{\theta}(x), y = 0)$

$$Cost(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & y = 1 \\ -\log(1 - h_{\theta}(x)) & y = 0 \end{cases}$$

- $Cost(h_{\theta}(x), y) \rightarrow 0$
 - When $h_{\theta}(x) \rightarrow 0$, i.e., predict $y = 0$
 - Good predication, low cost
- $Cost(h_{\theta}(x), y) \rightarrow \infty$
 - When $h_{\theta}(x) \rightarrow 1$, i.e., predict $y = 1$
 - Bad predication!
 - High cost represents penalty



Cost Function

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & y = 1 \\ -\log(1 - h_{\theta}(x)) & y = 0 \end{cases}$$

$$\text{Cost}(h_{\theta}(x), y) = -y \log h_{\theta}(x) - (1 - y) \log(1 - h_{\theta}(x))$$

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m \left[y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) \right]$$

Gradient Descent Algorithm

$$h_{\theta}(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m \left[y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)})) \right]$$

Repeat until convergence {
$$\theta_j = \theta_j - \alpha \frac{\partial J(\theta_0, \theta_1, \dots)}{\partial \theta_j}$$

}

$$\frac{\partial J(\theta_0, \theta_1, \dots)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

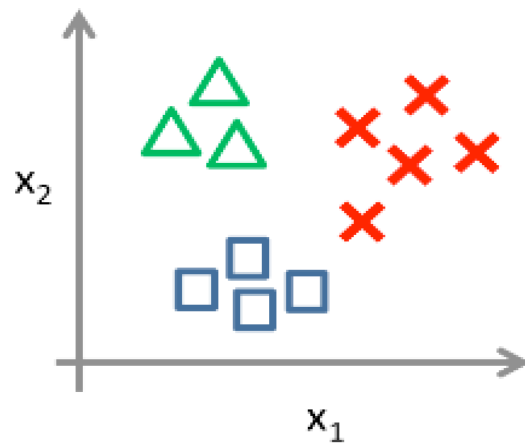
Looks identical to linear regression!




How about Multi-class Classification

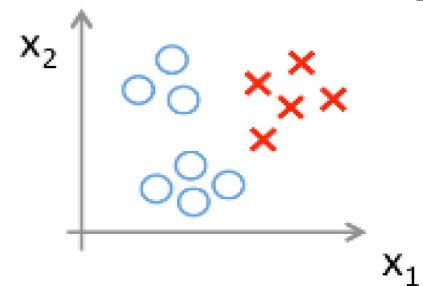
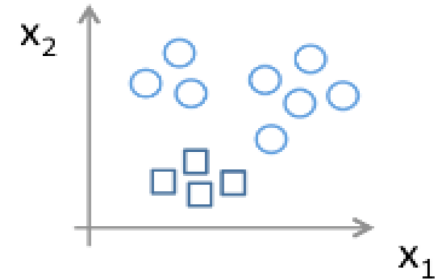
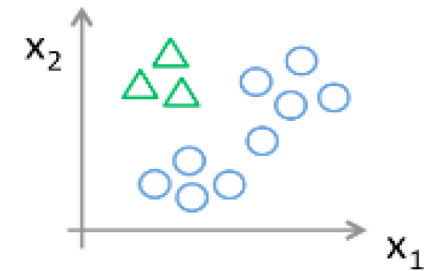
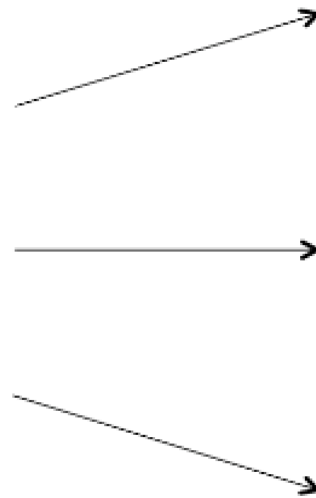
- Train a logistic regression classifier $h_{\theta}^{(i)}(x)$ for each class i to predict the probability of $y = 1$
- On a new input x , pick the class that maximizes $\max_i \left(h_{\theta}^{(i)}(x) \right)$

One-vs-All Approach

One-vs-all (one-vs-rest):



Class 1: 
Class 2: 
Class 3: 



Exercise

- Assume that there is a classification problem with 4 classes. Each instance has 5 features. What is the total number of parameters, if you are solving it by using linear logistic regression and one-vs-all approach? Remember to include θ_0 .
- We have 4 classes, so we need 4 binary classifiers. In each classifier, we have $\theta_0, \theta_1, \theta_2, \dots, \theta_5$. Thus, in total, we have $4 \times 6 = 24$ parameters.

Classification Metrics

- True Positives (**TP**): the actual class of the data point was **True** and the predicted is also **True**
- True Negatives (**TN**): the actual class of the data point was **False** and the predicted is also **False**
- False Positives (**FP**): the actual class of the data point was **False** and the predicted is **True**
- False Negatives (**FN**): the actual class of the data point was **True** and the predicted is **False**

		Actual	
		True	False
Predicted	True	TP	FP
	False	FN	TN

Accuracy

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

- A good measure when the target variable classes in the data are nearly balanced
 - 60% classes in our fruit images are apples and 40% are oranges
- NEVER used as a measure when the target variable classes in the data are a majority of one class (Why?)

		Actual	
		True	False
Predicted	True	TP	FP
	False	FN	TN

Limitation of Accuracy

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

- Example: In daily life, 5 people in 100 people have cancer.
- Consider a fake cancer detection model **only outputs 'health'**, its accuracy can achieve **95%**.
- Although its accuracy is good, is it a good model? NO.
- Observation: Accuracy performs bad when the target variable classes in the data are a majority of one class.

Precision and Recall

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Precision is about being precise; even if we managed to capture only one True case, and we captured it correctly, then we are 100% precise

		Actual	
		True	False
Predicted	True	TP	FP
	False	FN	TN

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Recall is not so much about capturing cases correctly but more about capturing all True cases

		Actual	
		True	False
Predicted	True	TP	FP
	False	FN	TN

F1 Score

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- A single score that represents both Precision and Recall
- F1 Score is the harmonic mean of Precision and Recall
- Different with arithmetic mean, harmonic mean is closer to the smaller number as compared to the larger number
- If Precision is 0.01 and Recall is 0.99, then arithmetic mean is 0.5 and harmonic mean is 0.0198
- Therefore, F1 score of the previous cancer detection model will be 0 (“positive” refers to having cancer)

Exercise

- Assume that there are 1,000 documents in total. Among them, 700 documents are related to big data analysis. You build a model to identify documents related to big data analysis. As a result, your model returns 800 documents, but only 550 of them are relevant to big data analysis. What is the recall of your model? What is the F1 score of your model?
- $TP = 550$. $FP = 250$. $FN = 150$
- $Precision = 550/800$.
- $Recall = 550/700$.
- $F1 = (2 * 550/800 * 550/700) / (550/800 + 550/700) = 0.7333333333$.