Week 4:
Logistic Regression

A Simple Example of Learning

- Each day you get lunch at a new cafeteria.
  - Your lunch consists of sandwich, chips, and candy bars.
  - You get several portions of each per meal.
  - The cashier only tells you the total price of the meal.
- you want to find out the total price for my future purchase.

Each meal price gives a linear constraint on the prices of the portions:

$$price = x_{sandwich} w_{sandwich} + x_{chips} w_{chips} + x_{candybar} w_{candybar}$$

Two ways to solve the equations

- The obvious approach: just to solve a set of simultaneous linear equations, one per meal
  - analytical solution with exact solution but costly.
- Or, use a numerical solution: making guesses at the solution and testing whether the problem is solved well enough to stop.

The prices of the portions are like the weights:

$$\mathbf{w} = (w_{sandwich}, w_{chips}, w_{candybar})$$

start with guesses for the weights and then adjust the guesses to give a better fit to the prices given by the cashier.

Adaline Review

- The output of an Adaline neuron: a **real value** output which is a weighted sum of its inputs:

Weight vector

$$\sum_i w_i x_i = \mathbf{w}^T \mathbf{x}$$

Input vector

- The aim of learning is to minimize the discrepancy between the predicted output and the actual output.
  - **How do we measure the discrepancies/errors/loss?**
  - **How do we update the weights?**

Batch Adaline Review

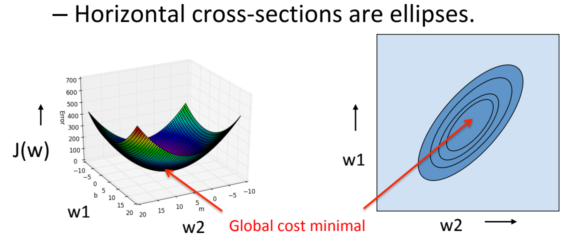
- Define the error as the Sum of Squared Errors over all training cases: $E = \frac{1}{2} \sum_n (y_n - \hat{y}_n)^2$
- Now differentiate to get error derivatives for weights:

$$\frac{\partial E}{\partial w_i} = \frac{1}{2} \sum_n \frac{\partial \hat{y}_n}{\partial w_i} \frac{\partial E_n}{\partial \hat{y}_n} = -\sum_n x_{i,n} (y_n - \hat{y}_n)$$

- The batch learning rule changes the weights in proportion to their error derivatives summed over all training sample. $\Delta w_i = -\varepsilon \frac{\partial E}{\partial w_i}$

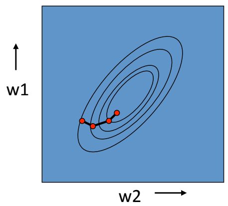
Error Surface

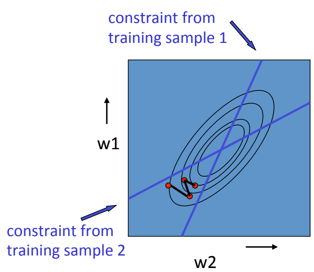
- For a linear neuron of two features, the error surface of the cost function lies in a space with a horizontal axis for each weight and one vertical axis for the error.
  - Vertical cross-sections are parabolas.
  - Horizontal cross-sections are ellipses.



Batch v.s. Stochastic

- Batch learning does steepest descent on the error surface
- Stochastic Gradient Descent (Online learning) zig-zags around the direction of steepest descent



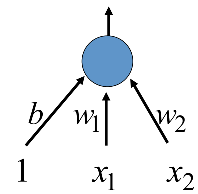


Bias

- The bias in an Adaline model makes it more flexible

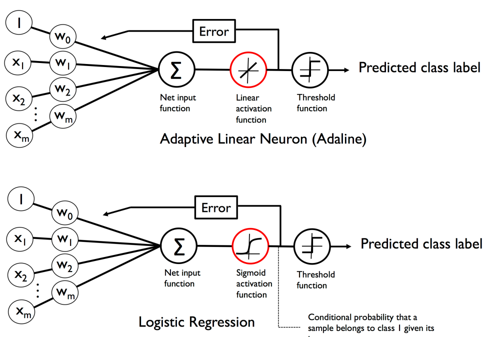
$$\hat{y} = b + \sum_i x_i w_i$$

- A bias is exactly equivalent to a weight on an extra input line that always has the input value of 1.



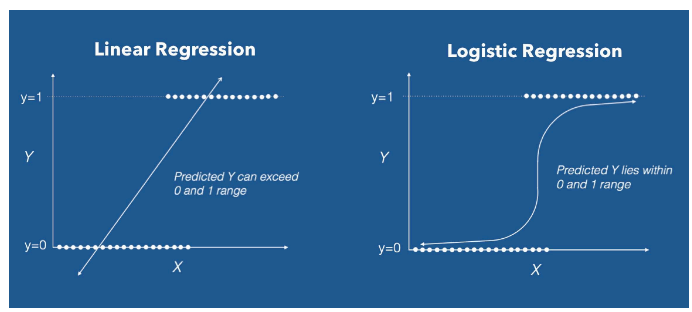
Activation Function

Adaline model uses the identify function **f(x) = x** as the activation function. But there are limitations…



Logistic Regression

- A better model to fit binary classification problems
- Simple idea: use a sigmoid activation function to squash the input into a logistic curve:
  the output values are always in the range of 0 to 1.



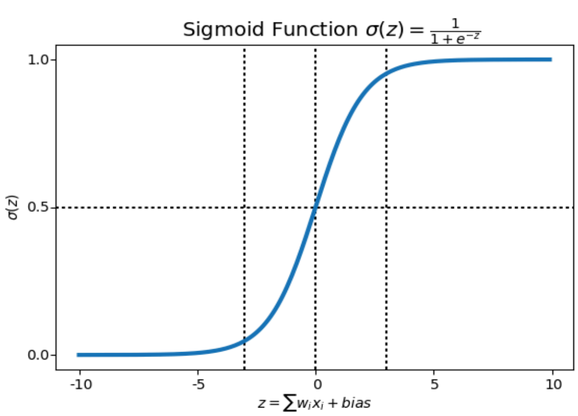


Image source: https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148

Cost Function for Logistic Regression

Logistic regression cost function

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

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

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

Taken from Prof. Andrew Ng.'s Coursera ML course

The goal is to maximize the likelihood of predicting the expected output.

Cost Function for Logistic Regression

Logistic regression cost function

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

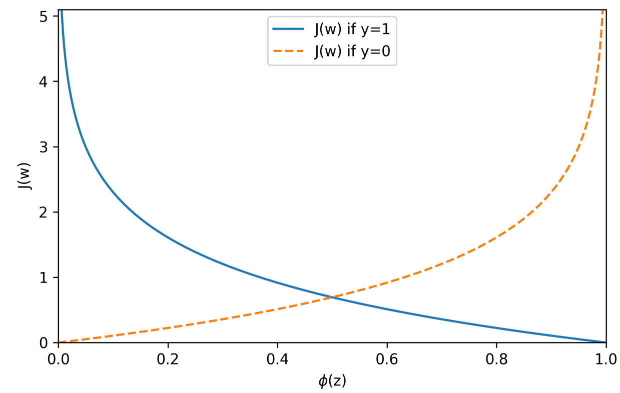
$$= -\frac{1}{m}\left[\sum_{i=1}^{m} \boxed{y^{(i)}\log h_\theta(x^{(i)})} + \boxed{(1-y^{(i)})\log(1-h_\theta(x^{(i)}))}\right]$$

0 If actual y=1

0 If actual y=0

$$\boxed{\begin{aligned} J(\theta) &= \frac{1}{m}\sum_{i=1}^{m}\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) \\ \mathrm{Cost}(h_\theta(x), y) &= \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1-h_\theta(x)) & \text{if } y = 0 \end{cases} \\ &\text{Note: } y = 0 \text{ or } 1 \text{ always} \end{aligned}}$$

Cost Function for Logistic Regression

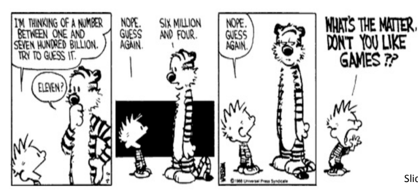


Exercise

- Compare the simple Python implementation for both the Adaline model and Logistic Regression model.

Learning Is Impossible

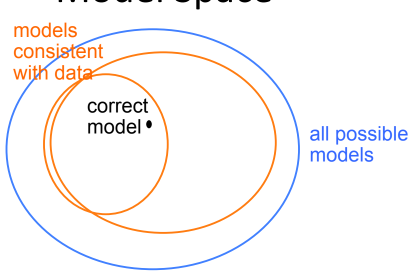
- What's my rule (model)?
  - 1 2 3 ⇒ satisfies rule
  - 4 5 6 ⇒ satisfies rule
  - 6 7 8 ⇒ satisfies rule
  - 9 2 31 ⇒ does not satisfy rule



- Possible rules (models)
  - 3 consecutive single digits
  - 3 consecutive integers
  - 3 numbers in ascending order
  - 3 numbers whose sum is less than 25
  - 3 numbers, each < 10
  - 1, 4, or 6 in first column
  - "yes" to first 3 sequences, "no" to all others

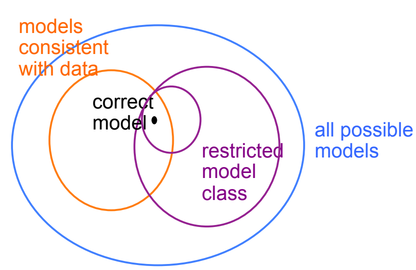
Slides adopted from Bias and Variance by Mike Mozer

Model Space



- More data helps

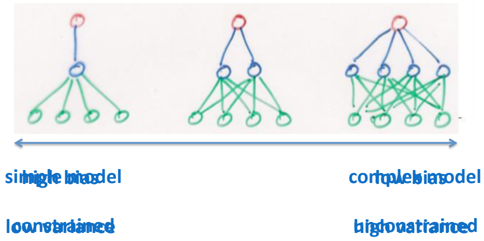
Model Space



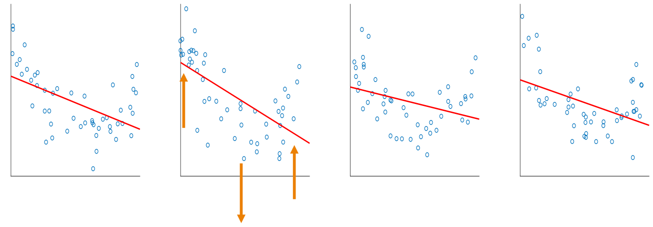
- Restricting model class can help
- Or it can hurt
- Depends on whether restrictions are appropriate

Selecting Models

- Models range in their flexibility to fit arbitrary data

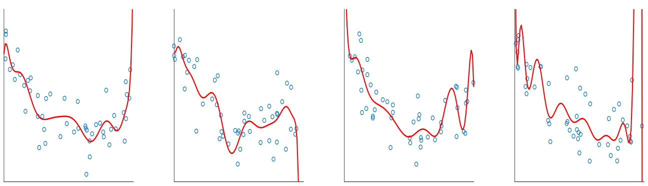


Bias

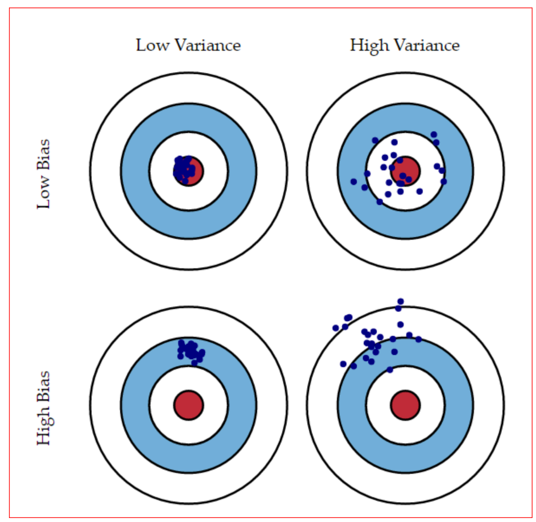


- Regardless of training sample, or size of training sample, model will produce consistent errors
- Models with high bias over-simplify the model.
- The problem of underfitting

Variance



- Different samples of training data yield different model fits
- Model with high variance pays a lot of attention to training data and does not generalize on unseen data.
- The problem of overfitting.



Use Regularization to Reduce Overfitting

Add a term to the cost function to penalize large weight values of model.

$$J(\boldsymbol{w}) = -\sum_i y^{(i)} \log\left(\phi\left(z^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - \phi\left(z^{(i)}\right)\right)$$

$$J(\boldsymbol{w}) = \sum_{i=1}^{n}\left[-y^{(i)} \log\left(\phi\left(z^{(i)}\right)\right) - \left(1 - y^{(i)}\right) \log\left(1 - \phi\left(z^{(i)}\right)\right)\right] + \boxed{\frac{\lambda}{2}\|w\|^2}$$

The hyper-parameter C in logistic regression model is the inverse-regularization parameter:
smaller values specify stronger regularization and thus smaller the weight coefficients.