1. Tìm đạo hàm của hàm Cost trong Unregularized Logistic Regression để phục vụ cho hàm Gradient Descent:

*Chú ý: Trong quá trình training, θ là biến số. Trong quá trình dự đoán, x là biến số.*

Đặt:

*J(θ) =*

*Chú ý:*

* *=*

Xét đạo hàm của theo , với j = 0,1,2,...n:

*= (1)*

*Chú ý:*

* *. Do*
* *= = = =*
* *= =*

*= =*

*= +*

*= . = .*

* *= (2)*
* *= (3)*

*(1), (2), (3)*

Kết luận:

Đạo hàm của theo , với j = 0,1,2,...n:

1. Phân tích tốc độ thay đổi các trọng số của mô hình trong quá trình thực thi Gradient Descent.

Công thức Gradient Descent trong Linear & Logistic Regression:

Mỗi , Với j = 0,1,2,...n:

= - α. = -

Đánh giá:

* Để hội tụ (Tìm được xấp xỉ cực tiểu), tốc độ thay đổi của các trọng số cần khác nhau do điểm xuất phát (Giá trị khởi tạo của các trọng số) là ngẫu nhiên.
* Thành phần gây ra sự khác biệt trong độ dài của mỗi bước nhảy () do các training example khác nhau.

1. Tìm đạo hàm của hàm Cost trong Neural Network để phục vụ cho hàm Gradient Descent.

**Trả lời:**

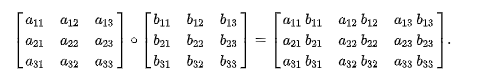
* 1. Kiến thức tiên quyết:
* Khoảng cách Mahattan (Mahattan distance):

d(x,y) = , m: Số chiều của vector (Vector dimension).

VD: Trong không gian Euclid 3 chiều, cho 2 vector x = (1;2;3), y = (4;5;6). Khoảng các Mahattan giữa 2 vector: d(x,y) = |1-4| + |2-5| + |3 - 6| = 9.

Chú ý: Mục đích sử dụng của Mahattan distance nằm ngoài phạm vi của Back propagation.

* L1 normalization (Dạng chuẩn L1, L1 norm):
  + Định nghĩa: Trong không gian Euclid, dạng chuẩn L1 của vector x là khoảng cách Manhattan giữa x và vector gốc tọa độ.
  + Công thức:
  + Ví dụ: Trong không gian Euclid 3 chiều, cho vector x = (1;2;3). |1| + |2| + |3| = 6
  + Ứng dụng: Thay thế cho ký hiệu .
* Phép nhân Hadamard (Hadamard product hay element – wise ):
  + Ký hiệu: AOB hoặc AB
  + Ví dụ:



* 1. Chứng minh:

Đặt:

*J(θ) =*

*+*

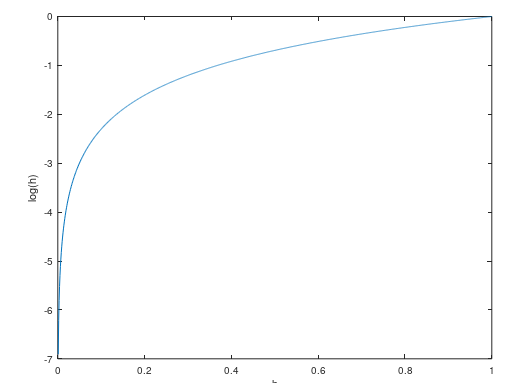
*+* C , C: thành phần Regularization (Regularization term).

*Chú ý:*

* *: Lớp Output không chứa thành phần Bias.*
* *: Regularization rate.*
* *: Không thực hiện chính quy hóa (Regularization) cho thành phần Bias (i = 0).*
* Xét một training example:

Do: (0;1)

* và (-∞; 0)



* và (0; +∞)

Ta có:

*J(θ) = +*

Chú ý: Thành phần Regularization chỉ sử dụng khi xét toàn bộ training set. Vì lúc đó, hệ số của các trọng số mới được tính toán đấy đủ.

, với

* Xét lớp Output:

= = , với z = ; L: Chỉ số của lớp Output; : Số lượng node của Output.

Tương tự như chứng minh Gradient Descent của Logistic Regression tại mục 1:

, với y =

Ta có: có kết quả là một vector ma trận, mà các ma trận trong đó có giá trị như sau:

Chú ý: Vector ma trận có tên gọi khác là ma trận 3 chiều.

=

=

...

=

Vậy:

Chú ý: Bản chất, đạo hàm giữa các ma trận là **tập hợp** **(liệt kê)** đạo hàm của mỗi phần từ của hai ma trận với nhau. VD: A có kích thước 2x3, B có kich thước 3x4, vậy có tổng cộng 72 phần tử. Ma trận chẳng qua là một cách để biểu diễn sự **liệt kê** đó. Cho nên, các phần tử trong ma trận sau khi đạo hàm có thể được sắp xếp tùy ý.

**Q&A:**

1. Overfitting có đồng nghĩa với việc độ chính xác giảm đi trong quá trình làm việc với tập dữ liệu mới không?

**Trả lời:**

Không. Tùy vào từng bộ dữ liệu cũng như từng bài toán.

1. Định nghĩa & phân loại Linear Regression.

**Trả lời:**

Trong thống kê, một mô hình hồi quy (Regression model) được xem là tuyến tính (Linear) khi tất cả các số hạng (Terms) trong mô hình đó là một trong hai dạng sau:

* Hằng số (Constant).
* Tích của một tham số (Parameter) và một biến độc lập (Independent variable).

VD: Một mô hình Multivariate polynomial regression như sau:

*= +* + + + + 99

Trong đó:

* , với i = 0,1,2,3,4, là các tham số hình thức (Paramters).
* , với i = 0,1,2,3,4, là các biến độc lập.
* là hằng số.

Như vậy, Linear regression **không** chỉ là các mô hình mà đồ thị của chúng là tuyến tính (đường thằng - line, mặt phẳng - plane, siêu mặt phẳng – hyperplane) mà bao gồm cả các mô hình mà đồ thị có dạng phi tuyến.

1. Back propagation dùng để làm gì?

**Trả lời:**

Hỗ trợ tính toán các đạo hàm phức tạp trong quá trình Gradient Descent. Bản chất, Back propagation sử dụng Chain rules và tính toán theo chiều từ lớp ngoài cùng trở vào trong.

Kết luận:

Back propagation = Chain rules + Tính toán từ lớp ngoài cùng trở vào trong.