Table of Contents

[Prevent Overfitting: Regularization 2](#_Toc29204045)

[Regularized Linear Regression 2](#_Toc29204046)

[Regularized Logistic Regression 4](#_Toc29204047)

# **Prevent Overfitting: Regularization**

Underfitting: hypothesis does not fit training data well

(because hypothesis is too simple or too few features)

Overfitting: hypothesis fits the training set well but does not generalize to predict new data.

(because of a complicated function that creates many unnecessary curves and angles unrelated to the data)

A close up of a map

Description automatically generated

overfitting

underfitting

Address overfitting:

1). Drop some features

* manually select
* model selection algorithm

2). Regularization:

Keep all features, but reduce their weights (i.e. use small parameters ) by inflating their costs

🡪 simpler hypothesis function

🡪 less prone to overfitting

Works well when we have a lot of features, and each contributes a bit to predicting y.

## **Regularized Linear Regression**

Add a regularization term to cost function that penalizes parameters for being large.

Cost function: (not regularize )

* By convention, we don’t penalize , so j starts from 1 rather than 0

n = number of features j = 0, ... n

m = size of training set i = 1, ... m

* : regularization parameter, determine how much the costs of are inflated

A trade-off between fitting training data well and keeping parameters small

If is too large: penalize too hard, will set all parameters to be 0, , underfitting

If is too small: overfitting

A close up of text on a white background

Description automatically generatedA close up of text on a white background

Description automatically generated

A close up of text on a white background

Description automatically generated

Gradient descent for regularized linear regression:

Repeat {

simultaneously update all for j = 1, ..., n

}

Note that and the second term is the same as before

i.e. we reduce the value of by some amount on every update:

Normal equation for regularized linear regression:

Design matrix: m \* n+1 matrix, each row is ith training example

m-dimensional vector

is a (n + 1) \* (n + 1) matrix with 0, 1....1 on the diagonal and 0 elsewhere

When m < n, is non-invertible, but regularization solves non-invertibility:

If > 0, is invertible.

## **Regularized Logistic Regression**

Cost function: (not penalize )

* By convention, we don’t penalize , so j starts from 1 rather than 0

n = number of features j = 0, ... n

m = size of training set i = 1, ... m

Gradient descent for regularized logistic regression:

Same algorithm as regularized linear regression but different hypothesis function

Repeat {

simultaneously update all for j = 1, ..., n

}

|  |
| --- |
| function [J, grad] = costFunctionReg(theta, X, y, lambda)  % Compute cost and gradient of the cost w.r.t. to theta parameters  % for regularized logistic regression  m = length(y);  % We do not regularize paratemter theta(0)  % i.e. theta(1) as index starts from 1 in MATLAB  thetaForReg = theta;  thetaForReg(1) = 0;  % return cost  h = sigmoid(X \* theta);  J = -(1/m) \* (y' \* log(h) + (1-y)' \* log(1 - h)) + (lambda/(2\*m)) \* sum(thetaForReg .^2);  % return gradient  grad = (1/m) \* X' \* (h - y) + (lambda / m) \* thetaForReg;  end |

Advanced optimization for regularized logistic regression:

|  |
| --- |
| % As X1 and X2 are not linearly related, we add polynomial terms  % to create a more complex hypothesis function and decision boundary.  % However, a complicated hypothesis function may overfit the data,  % so we implement regularization to prevent overfitting.  % Add Polynomial Features  % mapFeature also adds a column of ones, so the intercept term is handled  X = mapFeature(X(:,1), X(:,2));  % Initialize fitting parameters  initial\_theta = zeros(size(X, 2), 1);  % Set regularization parameter lambda to 1  % Try different values of lambda (0, 1, 10, 100)  lambda = 1;  % Optimize  options = optimset('GradObj', 'on', 'MaxIter', 400);  [theta, J, exit\_flag] = fminunc(@(t)(costFunctionReg(t, X, y, lambda)), initial\_theta, options); |