Logistic Regression and Regularization

Data Science Decal

Hosted by Machine Learning at Berkeley

Agenda

Background

Broadening Regression

Algorithms

Multinomial Regression

Regularization

Questions

Background

Regression (Review From Last Week)

- **Regression** models are designed to make predictions of **dependent** variables from **explanatory** variables.

Regression (Review From Last Week)

- **Regression** models are designed to make predictions of **dependent** variables from **explanatory** variables.
- **Linear** Regression fits a model for explanatory variable x with d dimensions as

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_d x_d$$

Regression (Review From Last Week)

- **Regression** models are designed to make predictions of **dependent** variables from **explanatory** variables.
- **Linear** Regression fits a model for explanatory variable x with d dimensions as

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_d x_d$$

- Linear regression optimizes over the average **Mean Squared Error** over n points:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - h_\theta(x^{(i)})^2$$

Important Question

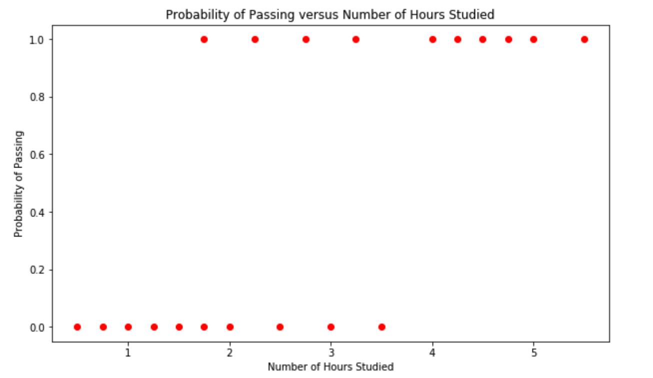
- You have a final exam coming up in a P/NP class. How much should you study so that you have at least a 90% chance of passing?

Important Question

- You have a final exam coming up in a P/NP class. How much should you study so that you have at least a 90% chance of passing?
- You have a dataset that details the **number of hours** a person studied and **whether they passed**.

Fit This

Why is linear regression a sub-optimal algorithm for this problem?



Broadening Regression

A Different Sort of Regression

- Recall that linear regression is for **continuous** dependent variables, e.g. height vs. weight.

A Different Sort of Regression

- Recall that linear regression is for **continuous** dependent variables, e.g. height vs. weight.
- The current problem has **categorical** values for the dependent variable, i.e. P/NP - only two **classes**

A Different Sort of Regression

- Recall that linear regression is for **continuous** dependent variables, e.g. height vs. weight.
- The current problem has **categorical** values for the dependent variable, i.e. P/NP - only two **classes**
- We are regressing on (the likelihood of) **membership to a class**.

Same Problem, Different Faces

Some examples to help build intuition on the characteristics of these regression problems

Same Problem, Different Faces

Some examples to help build intuition on the characteristics of these regression problems

- Given age, body temperature, blood pressure, platelet count, does the subject have malaria?

Same Problem, Different Faces

Some examples to help build intuition on the characteristics of these regression problems

- Given age, body temperature, blood pressure, platelet count, does the subject have malaria?
- Given an image (a vector of pixel intensities), is there a cat depicted in the picture?

Same Problem, Different Faces

Some examples to help build intuition on the characteristics of these regression problems

- Given age, body temperature, blood pressure, platelet count, does the subject have malaria?
- Given an image (a vector of pixel intensities), is there a cat depicted in the picture?
- Given gender, age, major, does the person use Windows, Mac or Linux? (Note that this problem has three potential categories).

The Logistic Function

We introduce the logistic function, also called the **sigmoid**:

$$s(x) = \frac{1}{1 + e^{-x}}$$

The Logistic Function

We introduce the logistic function, also called the **sigmoid**:

$$s(x) = \frac{1}{1 + e^{-x}}$$

- What is its domain and range?

The Logistic Function

We introduce the logistic function, also called the **sigmoid**:

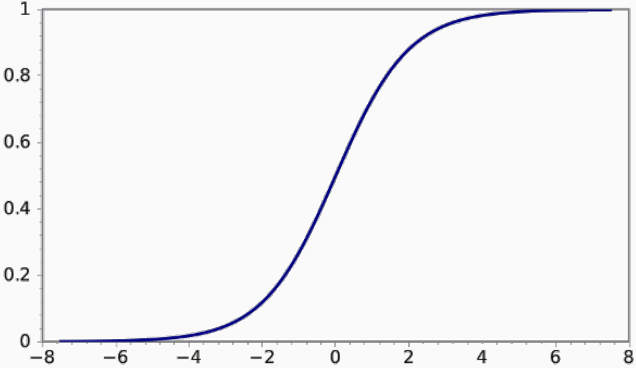
$$s(x) = \frac{1}{1 + e^{-x}}$$

- What is its domain and range?
- What is an interesting property regarding $s(x = 0)$?

Logistic Function: Dissected

Here's a picture. Notice that $s(x) - \frac{1}{2}$ intuitively appears to be an odd function. Hence, we have the following property for all x:

$$s(x) + s(-x) = 1$$



Why?

Recall from Linear Regression, that our hypothesis has the form:

$$h_\theta(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x$$

And, in our new problems, our dependent variables y are in $\{0, 1\}$.

Why?

Recall from Linear Regression, that our hypothesis has the form:

$$h_\theta(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x$$

And, in our new problems, our dependent variables y are in $\{0, 1\}$.
So, we propose that the hypothesis for Logistic Regression be:

$$h_\theta(x) = s\left(\sum_{i=0}^{d} \theta_i x_i\right) = s(\theta^T x)$$

Sanity Check

- The range of $h_\theta(x)$ is $(0, 1)$. This is good – we round in order to classify.

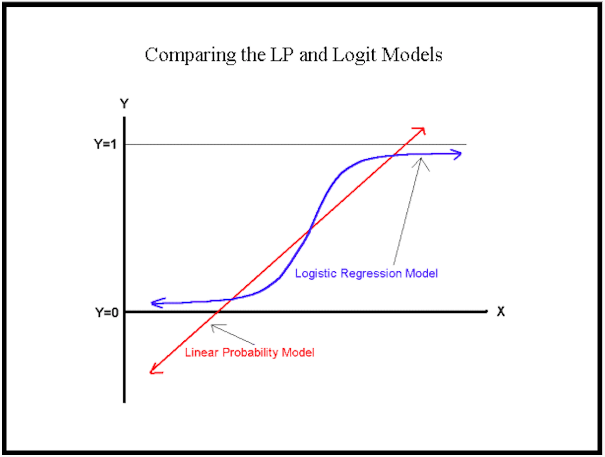
Sanity Check

- The range of $h_\theta(x)$ is $(0, 1)$. This is good – we round in order to classify.
- $h_\theta(x) + h_\theta(-x) = 1 \ \forall x$. Our model dictates that the probability of membership to one of the two classes is 1. This is good too.

Sanity Check

- The range of $h_\theta(x)$ is $(0, 1)$. This is good – we round in order to classify.
- $h_\theta(x) + h_\theta(-x) = 1 \ \forall x$. Our model dictates that the probability of membership to one of the two classes is 1. This is good too.
- Finally, we note that $h_\theta(x)$ is also known as an **expit**. It converts log-probability into a probability. The inverse function is known as a **logit**.

The Role of Sigmoid



Algorithms

The Mission

Our hypothesis is still parameterized by $\theta : [\theta_0 ... \theta_d]$.

The Mission

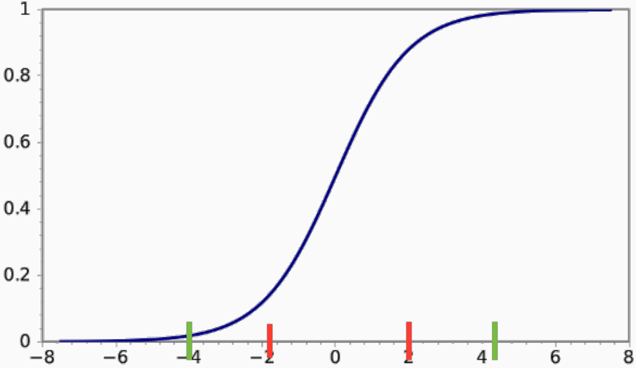
Our hypothesis is still parameterized by $\theta : [\theta_0 ... \theta_d]$.
The ideal θ would be such that for all x such that $y = 1$, we have

$$h_\theta(x) \approx 1 \rightarrow \theta^T x >> 0$$

A similar conclusion follows for $y = 0$.

The Mission: Visualized

Where would **ideal** values of $\theta^T x$ fall on the x-axis relative to the colored bands?



Behold: The Logistic Loss Function

To improve our weights, we will seek to minimize the Logistic Loss Function, with $z = h_\theta(x)$:

$$J(\theta) = -\sum_{i=1}^{n} y_i \dot{\ln} z_i + (1 - y_i) \dot{\ln} (1 - z_i)$$

Behold: The Logistic Loss Function

To improve our weights, we will seek to minimize the Logistic Loss Function, with $z = h_\theta(x)$:

$$J(\theta) = -\sum_{i=1}^{n} y_i \ln z_i + (1 - y_i) \ln(1 - z_i)$$

Because $\forall i, y_i \in \{0, 1\}$, only one of the addends is nonzero. That addend will be minimized when the expression involving z_i is closest to zero. (Remember $\ln(1) = 0$).

Gradient Descent Part 1

First, we arrange all the data points into a **design matrix**, X, where point $x^{(i)}$ is the i^{th} row: X_i. We then find the appropriate gradient to solve $\min_\theta J(\theta)$.

First, we arrange all the data points into a **design matrix**, X, where point $x^{(i)}$ is the i^{th} row: X_i. We then find the appropriate gradient to solve $\min_\theta J(\theta)$.

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^{n} -\frac{\partial z_i}{\partial \theta} \cdot \frac{\partial}{\partial z_i} J(\theta)$$

Gradient Descent Part 1

First, we arrange all the data points into a **design matrix**, X, where point $x^{(i)}$ is the i^{th} row: X_i. We then find the appropriate gradient to solve $\min_\theta J(\theta)$.

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^{n} -\frac{\partial z_i}{\partial \theta} \cdot \frac{\partial}{\partial z_i} J(\theta)$$

The first half of the chain rule:

$$\frac{\partial z_i}{\partial \theta} = \frac{\partial}{\partial \theta} s(\theta^T X_i) = s(\theta^T X_i)(1 - s(\theta^T X_i))X_i = z_i(1 - z_i)X_i$$

Gradient Descent Part 1

First, we arrange all the data points into a **design matrix**, X, where point $x^{(i)}$ is the i^{th} row: X_i. We then find the appropriate gradient to solve $\min_\theta J(\theta)$.

$$\frac{\partial J(\theta)}{\partial \theta} = \sum_{i=1}^{n} -\frac{\partial z_i}{\partial \theta} \cdot \frac{\partial}{\partial z_i} J(\theta)$$

The first half of the chain rule:

$$\frac{\partial z_i}{\partial \theta} = \frac{\partial}{\partial \theta} s(\theta^T X_i) = s(\theta^T X_i)(1 - s(\theta^T X_i))X_i = z_i(1 - z_i)X_i$$

You may want to check for yourself that $s'(x) = s(x)(1 - s(x))$

Gradient Descent Part 2

The other half of the chain rule:

$$\frac{\partial}{\partial z_i} J(\theta) = \frac{\partial}{\partial z_i}(y^{(i)} \dot{\ln} z_i + (1 - y_i) \dot{\ln}(1 - z_i))$$

$$= \frac{y_i}{z_i} - \frac{1 - y_i}{1 - z_i}$$

Gradient Descent Part 2

The other half of the chain rule:

$$\frac{\partial}{\partial z_i} J(\theta) = \frac{\partial}{\partial z_i}(y^{(i)}\dot{\ln} z_i + (1 - y_i)\dot{\ln}(1 - z_i))$$

$$= \frac{y_i}{z_i} - \frac{1 - y_i}{1 - z_i}$$

Hence, we have:

$$\frac{\partial J(\theta)}{\partial \theta} = -\sum_{i=1}^{n} \left(\frac{y_i}{z_i} - \frac{1 - y_i}{1 - z_i}\right) z_i(1 - z_i)X_i = -\sum_{i=1}^{n} (y_i - z_i)X_i$$

Gradient Descent Part 2

The other half of the chain rule:

$$\frac{\partial}{\partial z_i} J(\theta) = \frac{\partial}{\partial z_i} (y^{(i)} \dot{\ln} z_i + (1 - y_i) \dot{\ln}(1 - z_i))$$

$$= \frac{y_i}{z_i} - \frac{1 - y_i}{1 - z_i}$$

Hence, we have:

$$\frac{\partial J(\theta)}{\partial \theta} = -\sum_{i=1}^{n} \left(\frac{y_i}{z_i} - \frac{1 - y_i}{1 - z_i} \right) z_i (1 - z_i) X_i = -\sum_{i=1}^{n} (y_i - z_i) X_i$$

Expressed in Matrix form:

$$\frac{\partial J(\theta)}{\partial \theta} = -X^T (y - s(X\theta))$$

Gradient Descent Update

The (mini-)batch gradient descent update rule then becomes:

$$\theta \leftarrow \theta + \epsilon X^T(y - s(X\theta))$$

Gradient Descent Update

The (mini-)batch gradient descent update rule then becomes:

$$\theta \leftarrow \theta + \epsilon X^T (y - s(X\theta))$$

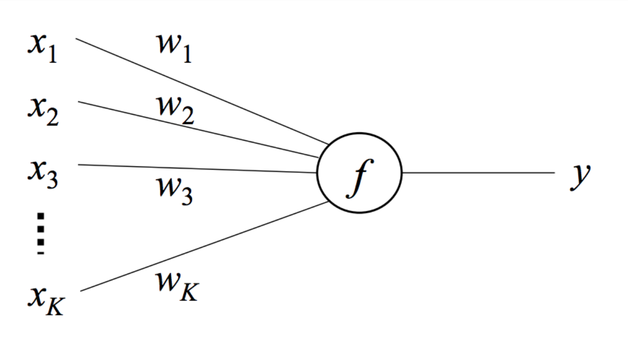
For stochastic gradient descent, the update is:

$$\theta \leftarrow \theta + \epsilon (y_i - s(X_i\theta))X_i$$

Take the time to understand how the summation is converted into a matrix calculation.

In fact, logistic regression models can be modeled as a one layer neural network!

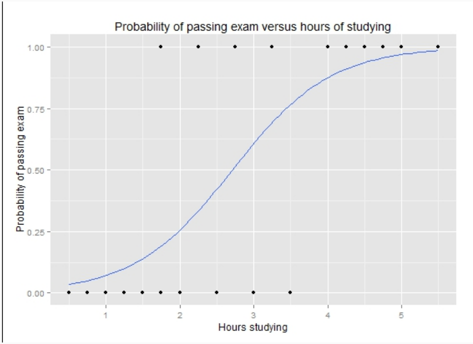
In fact, logistic regression models can be modeled as a one layer neural network!



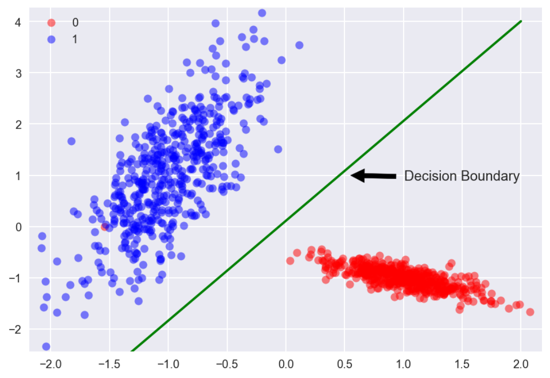
The edges comprise θ and the dot product has a sigmoid nonlinearity applied to it. Linear regression is the same model just without the sigmoid nonlinearity.

The End Result

Just for reference, here is the logistic regression solution to the initial problem that was posed:



Reminder: Linear Decision Boundary



Conceptual Summary: Linear vs. Logistic

Here we have some of the key distinctions between the two kinds of regression:

	Linear	Logistic
Label Type	Continuous	Categorical
Problem Type	Actual Regression	Actually Classification
Hypothesis	$\theta^T x$	$s(\theta^T x)$
Loss	Mean Squared	Logistic
Analytical Solution	Yes	No

Multinomial Regression

Multiple Choice

If logistic regression is answering True/False, then multinomial regression is answering a multiple choice question.

Multiple Choice

If logistic regression is answering True/False, then multinomial regression is answering a multiple choice question.

- We can frame all multiple choice questions as: "T/F: The answer is A. T/F: The answer is B. etc.

Multiple Choice

If logistic regression is answering True/False, then multinomial regression is answering a multiple choice question.

- We can frame all multiple choice questions as: "T/F: The answer is A. T/F: The answer is B. etc.
- The likelihood of any of the choices being correct forms a probability distribution.

Parameterizing Multinomial Regression

Earlier, having one vector θ was sufficient.

Parameterizing Multinomial Regression

Earlier, having one vector θ was sufficient.

- If we have possible classes: $c_1, ..., c_k$

Parameterizing Multinomial Regression

Earlier, having one vector θ was sufficient.

- If we have possible classes: $c_1, ..., c_k$
- Then we will need parameter vectors $\theta_1, ...\theta_k$.

Parameterizing Multinomial Regression

Earlier, having one vector θ was sufficient.

- If we have possible classes: $c_1, ..., c_k$
- Then we will need parameter vectors $\theta_1, ...\theta_k$.
- We can store these in parameter matrix: $\theta \in \mathcal{R}^{k \times d}$, with each parameter vector being a row θ_i

Calculating Actual Probability

We can give each class c_i a likelihood score:

$$\theta_{\mathbf{i}}\dot{x} \approx P(y = c_i)$$

Calculating Actual Probability

We can give each class c_i a likelihood score:

$$\theta_{\mathbf{i}}\dot{x} \approx P(y = c_i)$$

The product of the matrix θ and vector x results in a vector z which needs to be **normalized.** We used sigmoid before.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

The Softmax Function

Assuming that $h_\theta(x)$ still captures log-likelihood, we use the **softmax function** $S(z_i)$ to normalize multinomial regression scores:

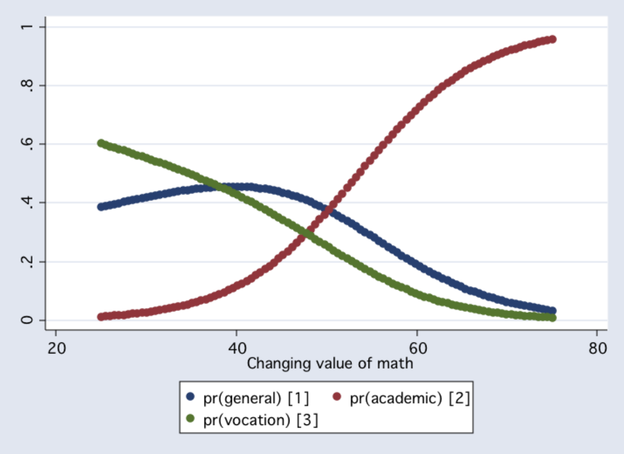
The Softmax Function

Assuming that $h_\theta(x)$ still captures log-likelihood, we use the **softmax function** $S(z_i)$ to normalize multinomial regression scores:

$$P(y = c_i) = S(z_i) = \frac{e^{\theta_i x}}{\sum_{j=1}^{k} e^{\theta_j x}}$$

Note the denominator is the sum of all the exponentiated scores in the vector z.

Softmax Probabilities



Softmax Loss and Gradient Update

The softmax loss function $J(\theta)$ is actually just a generalization of the logistic loss function:

$$J(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} 1\{y^{(i)} = c_j\} \ln(z_j)$$

Softmax Loss and Gradient Update

The softmax loss function $J(\theta)$ is actually just a generalization of the logistic loss function:

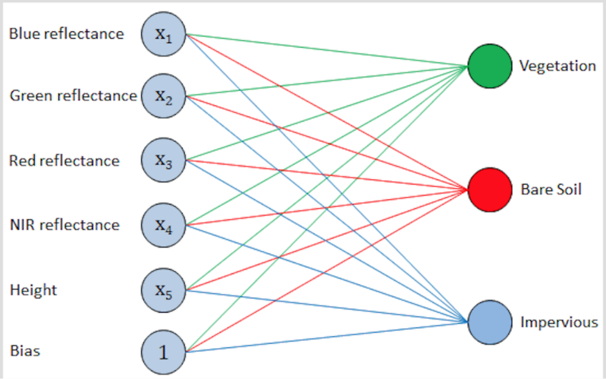
$$J(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} 1\{y^{(i)} = c_j\} \ln \left(z_j \right)$$

The gradient and gradient update are as follows:

$$\nabla_{\theta_i} J(\theta) = x(z_i - y_i)$$

$$\theta_i \leftarrow \theta_i - \epsilon x(z_i - y_i)$$

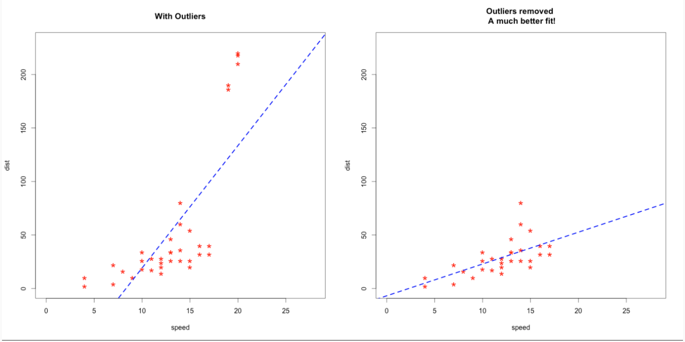
Softmax Result Diagram



Regularization

Possible Pitfalls

Regression models assume all the points come from the same distribution. However, real-world data comes with **outliers** and **noise**, which we want our models to see past.



Isolating the Issue

Our learned parameters θ very literally weight the different data components $[x_1 \ldots x_d]$. **Larger weights** mean **larger influence** of unwanted anomalies.

Isolating the Issue

Our learned parameters θ very literally weight the different data components $[x_1 \dots x_d]$. **Larger weights** mean **larger influence** of unwanted anomalies.

- Recall the magnitude of a vector v, also known as an **L2 norm** is computed as:

$$||v||_2 = \sqrt{\sum_{i=1}^{d} v_i^2}$$

Isolating the Issue

Our learned parameters θ very literally weight the different data components $[x_1 \ldots x_d]$. **Larger weights** mean **larger influence** of unwanted anomalies.

- Recall the magnitude of a vector v, also known as an **L2 norm** is computed as:

$$||v||_2 = \sqrt{\sum_{i=1}^{d} v_i^2}$$

- Another measurement of the magnitude, called the **L1 norm** is computed as:

$$||v||_1 = \sum_{i=1}^{d} |v_i|$$

The Regularized Regression Problem

With either norm, $L(\cdot)$ we can influence the optimization with regularization like so:

$$\min_{\theta} J(\theta) + \lambda L(\theta)$$

The Regularized Regression Problem

With either norm, $L(\cdot)$ we can influence the optimization with regularization like so:

$$\min_{\theta} J(\theta) + \lambda L(\theta)$$

λ is called the **regularization constant**, and decides how much to value norm-size versus regression error.

Modifying the Analytical Solution to Least Squares

In a previous lecture, we covered the normal equations solution to least squares regression. Now, we will do so for the regularized version:

$$\hat{\theta} = argmin_{\theta}||y - X\theta||_2^2 + \lambda||\theta||_2^2$$
$$= argmin_{\theta} y^T y - 2\theta^T X^T y + \theta^T X^T X\theta + \lambda\theta^T\theta$$

Modifying the Analytical Solution to Least Squares

In a previous lecture, we covered the normal equations solution to least squares regression. Now, we will do so for the regularized version:

$$\hat{\theta} = argmin_{\theta}||y - X\theta||_2^2 + \lambda||\theta||_2^2$$

$$= argmin_{\theta} y^T y - 2\theta^T X^T y + \theta^T X^T X\theta + \lambda\theta^T\theta$$

Taking the derivative and setting equal to zero is allowed since the function is convex in θ

$$-2X^T y + 2X^T X\hat{\theta} + 2\lambda\hat{\theta} = 0$$

$$(X^T X + \lambda I_d)\hat{\theta} = X^T y$$

$$\hat{\theta} = (X^T X + \lambda I_d)^{-1} X^T y$$

Regularized Gradient Descent Equation for Logistic Regression

The loss function gets an additional term, $\lambda |w|^2$, added to it. Just add its derivative to old derivative: The (mini-)batch gradient descent update rule then becomes:

$$\theta \leftarrow \theta + \epsilon \left(X^T (y - s(X\theta)) - 2\lambda w \right)$$

Regularized Gradient Descent Equation for Logistic Regression

The loss function gets an additional term, $\lambda|w|^2$, added to it. Just add its derivative to old derivative: The (mini-)batch gradient descent update rule then becomes:

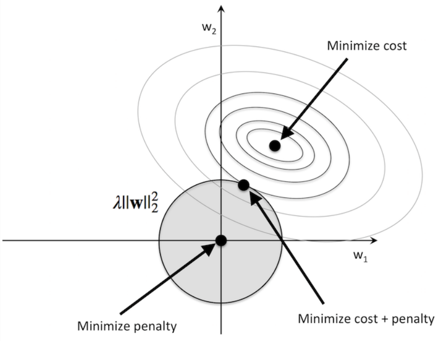
$$\theta \leftarrow \theta + \epsilon\left(X^T(y - s(X\theta)) - 2\lambda w\right)$$

For stochastic gradient descent, the update is:

$$\theta \leftarrow \theta + \epsilon\left((y_i - s(X_i\theta))X_i - 2\lambda w\right)$$

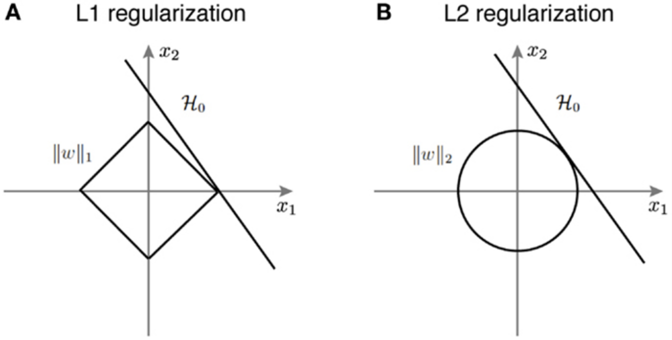
Visualizing Effects of Regularization

Regularization **prevents overfitting** by keeping the parameters from stretching out too far. The optimal point is an intersection of isocontours from the regular objective and the unit ball.

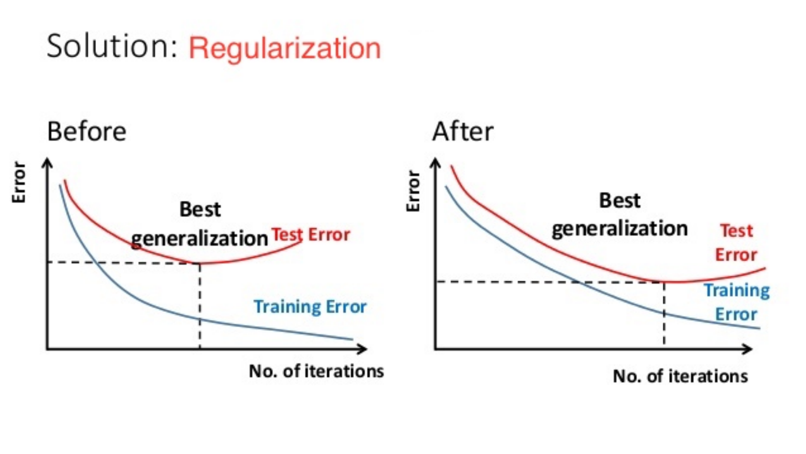


Visualizing the Effects of Different Norms

On the left is the unit L1 ball, and the right is the unit L2 ball. The L1 ball has vertices which are the likely points of tangency. This means Lasso regularization promotes **sparsity** in the optimal parameters.



Effect on Training, Test Error



Conclusion

- Regularization is a very easy way to improve generalization.

Conclusion

- Regularization is a very easy way to improve generalization.
- Variants of regularization have different effects.

Questions

Questions?