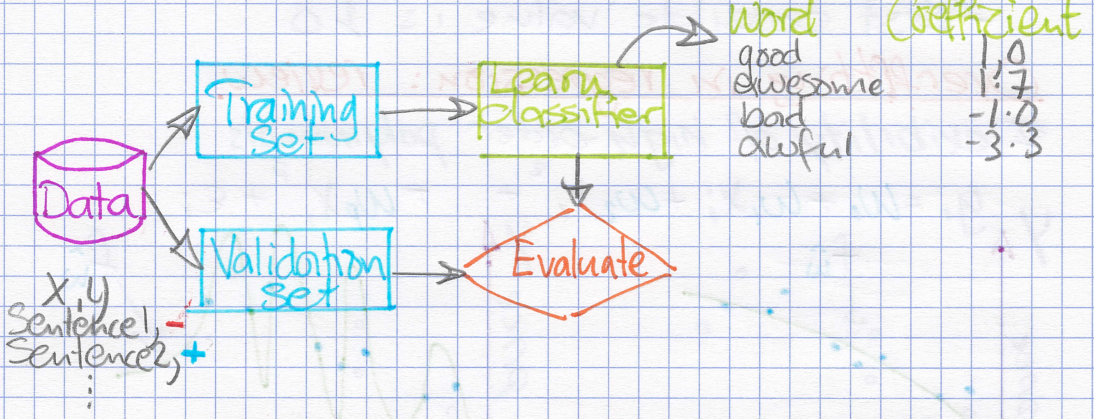
Overfitting & Regularization in Logistic Regression

Training and evaluating a classifier

Training a classifier = Learning the coefficients



Classification error

"Sushi was great"
"Food was OK"

Learned classifier

$\hat{y}$ = +, $\hat{y}$ = +

Test example
Sushi was great, +
Food was OK, -

Correct!
Mistake!
+, -
Hide label

Correct ↓
Mistakes ↓

Classification error & accuracy

- Error measures fraction of mistakes

$$\text{error} = \frac{\#\text{Mistakes}}{\text{Total number of data points}}$$

- Best possible value is 0.0

- often, measure accuracy
  - Fraction of correct predictions

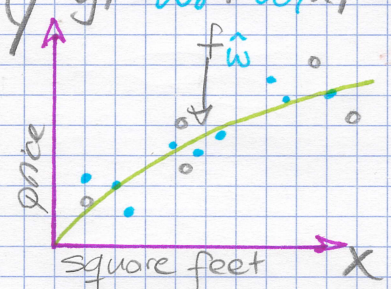
$$\text{accuracy} = \frac{\#\text{correct}}{\text{Total number of data points}}$$

  - Best possible value is 1.0

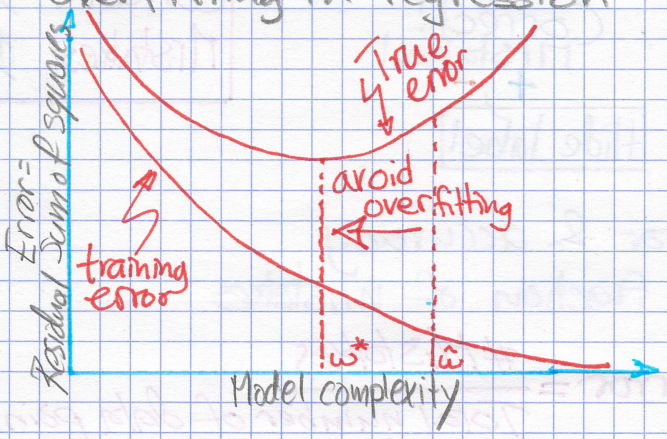
Overfitting in regression: review

Flexibility of high-order polynomials

$$y_i = w_0 + w_1 x_i + w_2 x_i^2 + \dots + w_p x_i^p + \varepsilon_i$$



Overfitting in regression



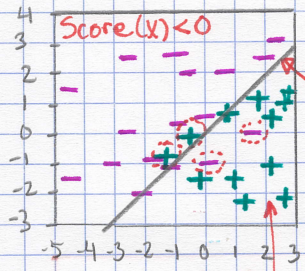
Overfitting if there exists w^*:

- training_error(w^*) > training_error($\hat{w}$)
- true_error(w^*) < true_error($\hat{w}$)

Overfitting in classification

Learned decision boundary

Feature	Value	Coefficient learned
$h_0(x)$	w_0 1	0.23
$h_1(x)$	w_1 x[1]	1.12
$h_2(x)$	w_2 x[2]	-1.07

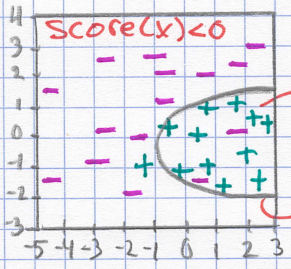


$0.23 + 1.12x[1] - 1.07x[2] = 0$

Score(x)>0

Quadratic features (in 2d)

Feature	Value	Coefficient learned
$h_0(x)$	1	1.68
$h_1(x)$	x[1]	1.39
$h_2(x)$	x[2]	-0.59
$h_3(x)$	$(x[1])^2$	-0.17
$h_4(x)$	$(x[2])^2$	-0.96



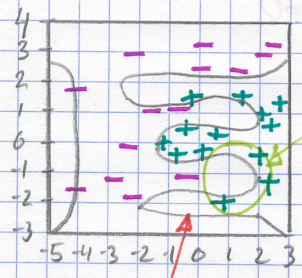
Score(x)>0

$1.68 + 1.39x[1] - 0.59x[2] - 0.17(x[1])^2 - 0.96(x[2])^2 = 0$

Degree 6 features (in 2d)

Feature	Value	Coefficients learned
$h_0(x)$	1	21.6
$h_1(x)$	$x[1]$	5.3
$h_2(x)$	$x[2]$	-42.7
$h_3(x)$	$(x[1])^2$	-15.9
⋮	⋮	⋮
$h_{11}(x)$	$(x[1])^6$	0.8
$h_{12}(x)$	$(x[2])^6$	-8.6

Coefficient values getting large

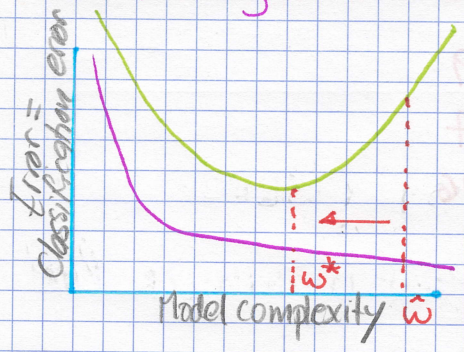


Score(x) < 0
⇒ $\hat{y} = -1$

extremely complex decision boundary

Often, overfitting associated with very large estimated coefficients $\hat{w}$

Overfitting in classification



Overfitting if there exists w^*

- training_error(w^*) > training_error($\hat{w}$)
- true_error(w^*) < true_error($\hat{w}$)

Overfitting in classifiers → Overconfident predictions

Logistic regression model

$P(y=+1 \mid x_i, w) = \text{sigmoid}(w^T h(x_i))$

The subtle (negative) consequence of overfitting in logistic regression

Overfitting → Large coefficient value

$\hat{w}^T h(x_i)$ is very positive (or very negative)

$\Rightarrow$ Sigmoid$(\hat{w}^T h(x_i))$ goes to 1 (or to 0)

Model becomes extremly overconfident of predictions

Penalizing large coefficients to mitigate overfitting

Desired total cost format

Want to balance:

i. How well function fits data

ii. Magnitude of coefficients

Total quality = measure of fit − measure of magnitude of coefficients

(want to balance)

measure of fit: (data likelihood) large # = good fit to training data

measure of magnitude of coefficients: large # = overfit

Maximum likelihood estimation (MLE):

Measure of fit = Data likelihood

- Choose coefficients w that maximize likelihood:

$$\prod_{i=1}^{N} P(y_i | x_i, w)$$

- Typically, we use the log of likelihood function (simplifies math and has better convergence properties)

$$\ell(w) = \ln \prod_{i=1}^{N} P(y_i | x_i, w)$$

Measure of magnitude of logistic regression coefficients

What summary # is indicative of size of logistic regression coefficients?

- Sum of squares (L_2 norm)

$$\|w\|_2^2 = w_0^2 + w_1^2 + w_2^2 + \ldots + w_D^2$$

- Sum of absolute value (L_1 norm)

$$\|w\|_1 = |w_0| + |w_1| + |w_2| + \ldots + |w_D|$$

Penalize large weights

Sparse solution

Consider specific total cost

$\max_w$ Total quality =

Measure of fit − measure of magnitude of coefficients

$\ell(w)$ ← log data likelihood

$\|w\|_2^2$ ← L2 penalty

L2 regularized logistic regression

Consider resulting objective

What if $\hat{w}$ selected to minimize:

$$\ell(w) - \lambda \|w\|_2^2$$

λ ← tuning parameter = balance of fit and magnitude

If $\lambda = 0$: Reduces $\max_w \ell(w)$ → Standard MLE solution

If $\lambda = \infty$: $\max_w \ell(w) - \infty \|w\|_2^2$ → Only care about penalizing w_j large coefficient → $w = 0$

If λ in between:

Balance data fit against the magnitude of the coefficients

L_2 regularized logistic regression

Pick λ using:

- Validation set (for large datasets)
- Cross-validation (for smaller datasets)

Bias-variance tradeoff

Large λ:

high bias, low variance

(e.g., $\hat{w} = 0$ for $\lambda = \infty$)

In essence, λ controls model complexity

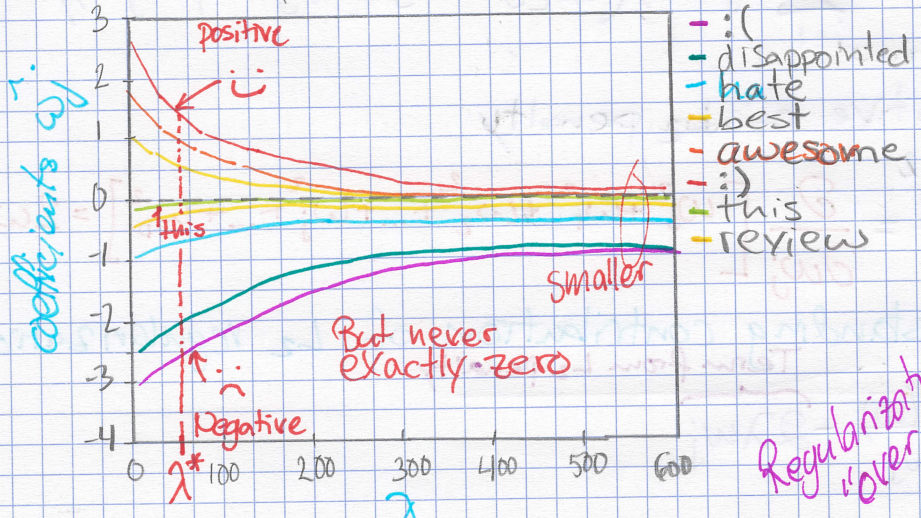
Small λ:

low bias, high variance

(e.g., maximum likelihood (MLE) fit of high-order polynomial for $\lambda = 0$)

Visualizing effect of L2 regularization in logistic regression

Coefficient path



Regularization reduces "overconfidence"

Finding best L_2 regularized linear classifier with gradient ascent

Algorithm:

while not converged:

$$w^{(t+1)} \leftarrow w^{(t)} + \eta \nabla \ell(w^{(t)})$$

need the gradient of regularized log likelihood

Gradient of L_2 regularized log-likelihood

Total quality = measure of fit − measure of magnitude of coefficients

$\ell(w)$ $\qquad$ $\lambda \|w\|_2^2$

$$\text{Total derivative} = \frac{\partial \ell(w)}{\partial w_j} - \lambda \frac{\partial \|w\|_2^2}{\partial w_j}$$

Derivative of (log-)likelihood

Sum over data points; feature value; difference between truth and prediction

$$\frac{\partial \ell(w)}{\partial w_j} = \sum_{i=1}^{N} h_j(x_i)\left(\mathbb{1}[y_i = +1] - P(y=+1 \mid x_i, w)\right)$$

Derivative of L_2 penalty

$$\frac{\partial \|w\|_2^2}{\partial w_j} = \frac{\partial}{\partial w_j}\left[w_0^2 + w_1^2 + w_2^2 + \ldots + w_j^2 + \ldots + w_D^2\right] = 2w_j$$

Understanding contribution of L_2 regularization

Term from L_2 penalty

$$\frac{\partial \ell(w)}{\partial w_j} - 2\lambda w_j$$

	$-2\lambda w_j$	Impact on w_j
$w_j > 0$	< 0	decreases $w_j \Rightarrow w_j$ becomes closer to 0
$w_j < 0$	> 0	increasing $w_j \Rightarrow w_j$ becomes closer to 0

Summary of gradient ascent for logistic regression with L_2 regularization

init $w_j^{(1)} = 0$ (or randomly, or smartly), $t = 1$

while not converged:

for $j = 0, \ldots, D$:

$$\text{partial}[j] = \sum_{i=1}^{N} h_j(x_i)\left(\mathbb{1}[y_i = +1] - P(y = +1 \mid x_i, w^{(t)})\right)$$

$$w_j^{(t+1)} \leftarrow w_j^{(t)} + \eta\left(\text{partial}[j] - 2\lambda w_j^{(t)}\right)$$

$$t \leftarrow t + 1$$

η: stepsize; $\frac{\partial \ell(w)}{\partial w_j}$: partial; $-2\lambda w_j^{(t)}$: only change!

Sparse logistic regression with L_1 regularization

Recall sparsity (many $\hat{w}_j = 0$) gives efficiency and interpretability

Efficiency

- If size(w) = 100B, each prediction is expensive
- If $\hat{w}$ sparse (many zeros), computation only depends on # of non-zeros

$$\hat{y}_i = \text{sign}\left(\sum_{\hat{w}_j \neq 0} \hat{w}_j h_j(x_i)\right)$$

Interpretability:
- Which features are relevant for prediction?

Sparse logistic regression

Total quality = measure of fit − measure of magnitude of coefficients

$\ell(w)$ $\quad$ $\|w\|_1 = |w_0| + \ldots + |w_D|$

L1 regularized logistic regression

Leads to sparse solutions!

L1 regularized logistic regression

Just like L2 regularization, solution is governed by a continuous parameter λ

$$\ell(w) - \lambda \|w\|_1$$

tuning parameter = balance of fit and sparsity

If $\lambda = 0$: No regularization → standard MLE solution

If $\lambda = \infty$: all weight is on regularization → $\hat{w} = 0$

If λ in between: Sparse solutions:

Some $\hat{w}_j \neq 0$, many other $\hat{w}_j = 0$

Regularization path – L1 penalty