Classification: Analyzing Sentiment

CS 229: Machine Learning

Emily Fox
Stanford University

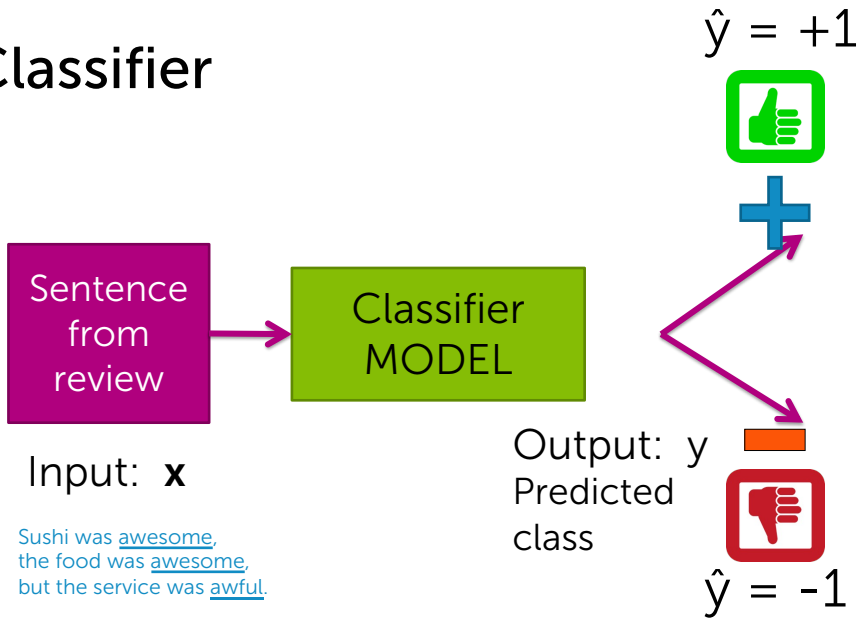
January 24, 2024

1

Classification

2

Classifier



3

Linear classifiers

4

Feature	Coefficient
…	…

Simple linear classifier

*Score(**x**)* = weighted sum of features of sentence

Sentence from review → Input: **x**

If *Score (**x**) > 0*:

$\hat{y}$ = +1

Else:

$\hat{y}$ = -1

5

A simple example: Word counts

Feature	Coefficient
good	1.0
great	1.2
awesome	1.7
bad	-1.0
terrible	-2.1
awful	-3.3
restaurant, the, we, where, …	0.0
…	…

Input $\mathbf{x}_i$:

Sushi was great,
the food was awesome,
but the service was terrible

word counts

$Score(x_i) = 1.2(1) + 1.7(1) - 2.1(1)$

$= 0.8 > 0$

$\Rightarrow \hat{y}_i = +1$

positive review

Called a linear classifier, because score is weighted sum of features.

6

More generically…

Model: $\hat{y}_i = \text{sign}(\text{Score}(\mathbf{x}_i))$

$$\text{Score}(\mathbf{x}_i) = w_0\, h_0(\mathbf{x}_i) + w_1\, h_1(\mathbf{x}_i) + \ldots + w_D\, h_D(\mathbf{x}_i)$$

$$= \sum_{j=0}^{D} w_j\, h_j(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{h}(\mathbf{x}_i)$$

sign(pos #) = +1
sign(neg #) = -1

feature 1 = $h_0(\mathbf{x})$ … e.g., 1

feature 2 = $h_1(\mathbf{x})$ … e.g., $\mathbf{x}[1]$ = #awesome

feature 3 = $h_2(\mathbf{x})$ … e.g., $\mathbf{x}[2]$ = #awful
or, log($\mathbf{x}[7]$) $\mathbf{x}[2]$ = log(#bad) x #awful
or, tf-idf("awful")

…

feature D+1 = $h_D(\mathbf{x})$ … some other function of $\mathbf{x}[1],\ldots, \mathbf{x}[d]$

7

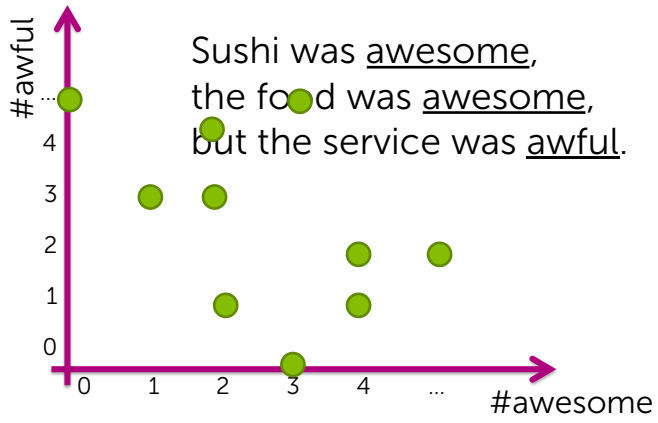
Decision boundaries

8

Suppose only two words had non-zero coefficient

Input	Coefficient	Value
	w_0	0.0
#awesome	w_1	1.0
#awful	w_2	-1.5

Score(x) = 1.0 #awesome – 1.5 #awful



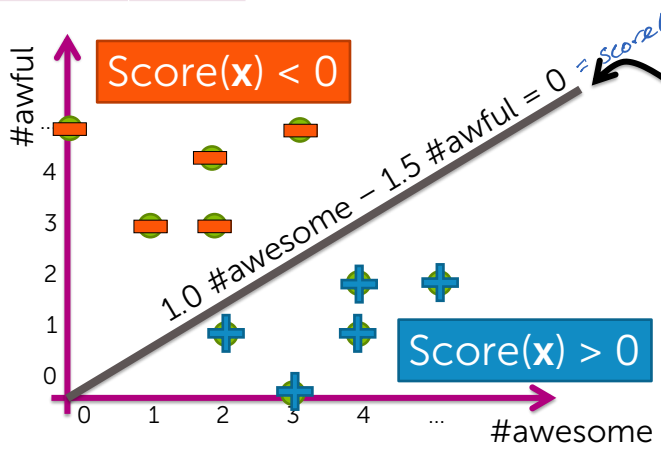
Sushi was awesome,
the food was awesome,
but the service was awful.

9

Decision boundary example

Input	Coefficient	Value
	w_0	0.0
#awesome	w_1	1.0
#awful	w_2	-1.5

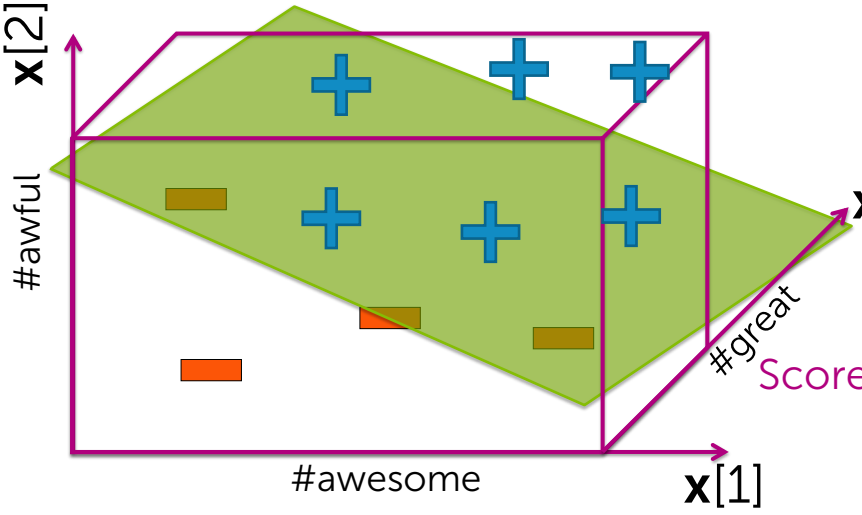
Score(x) = 1.0 #awesome – 1.5 #awful



Decision boundary separates + and – predictions

10

For more inputs (linear features)…

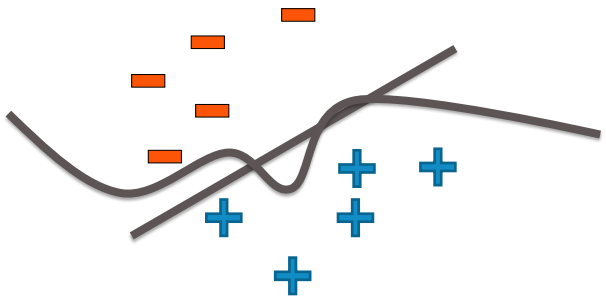


Score(**x**) = w_0
+ w_1 #awesome
+ w_2 #awful
+ w_3 #great

11

For general features…

For more general classifiers (not just linear features)
➔ more complicated shapes



12

Linear classifiers: Logistic regression

CS 229: Machine Learning

Emily Fox
Stanford University

January 24, 2024

13

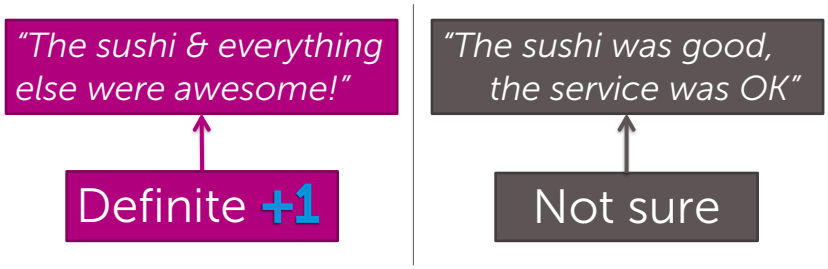
Are you sure about the prediction?

Class probability

14

How confident is your prediction?

- Thus far, we've outputted a prediction **+1** or **-1**
- But, how sure are you about the prediction?

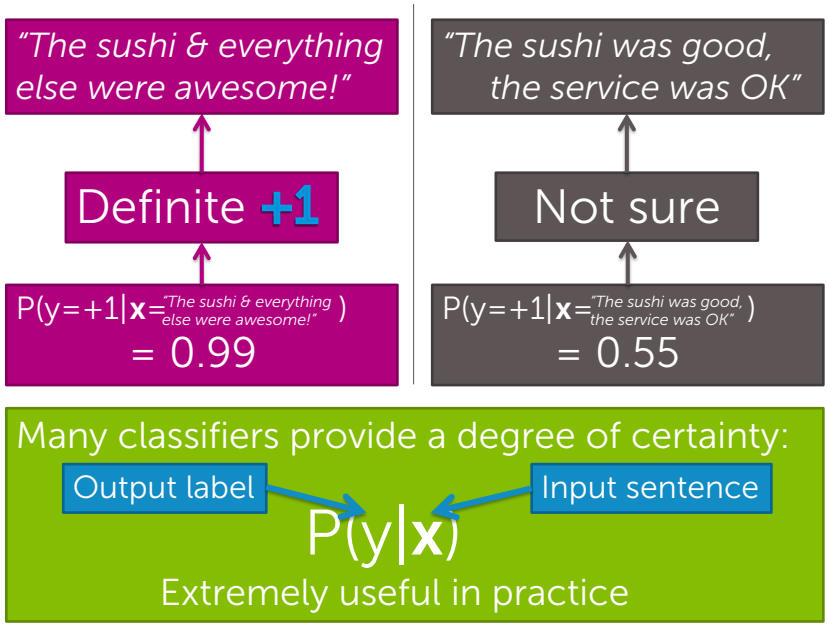


15

Using probabilities in classification

16

How confident is your prediction?



17

Goal: Learn conditional probabilities from data

Training data: N observations $(\mathbf{x}_i, y_i)$

x[1] = #awesome	x[2] = #awful	y = sentiment
2	1	+1
0	2	-1
3	3	-1
4	1	+1
…	…	…

Optimize **quality metric** on training data

Find best model $\hat{P}$ by finding best $\hat{w}$

Useful for predicting $\hat{y}$

18

Sentence from review

Input: **x**

Predict most likely class

$\hat{P}(y|x)$ = estimate of class probabilities

If $\hat{P}(y=+1|x) > 0.5$:

$\hat{y} = +1$

Else:

$\hat{y} = -1$

Estimating $\hat{P}(y|x)$ improves **interpretability**:

– Predict $\hat{y} = +1$ **and** tell me how sure you are

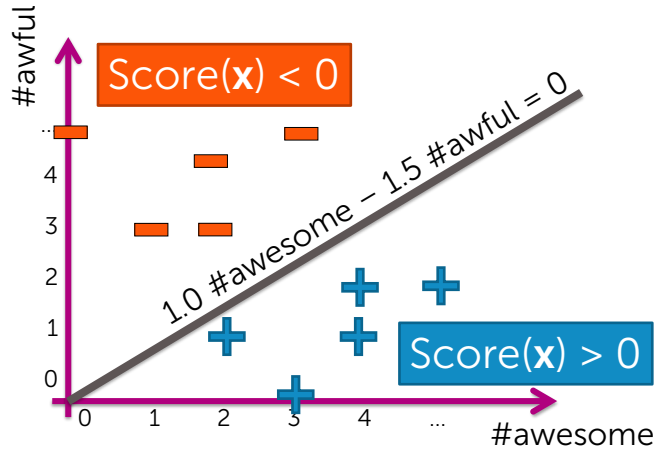
19

Predicting class probabilities with logistic regression

20

Thus far, we focused on decision boundaries

$$\text{Score}(\mathbf{x}_i) = w_0\, h_0(\mathbf{x}_i) + w_1\, h_1(\mathbf{x}_i) + \ldots + w_D\, h_D(\mathbf{x}_i)$$
$$= \mathbf{w}^T h(\mathbf{x}_i)$$

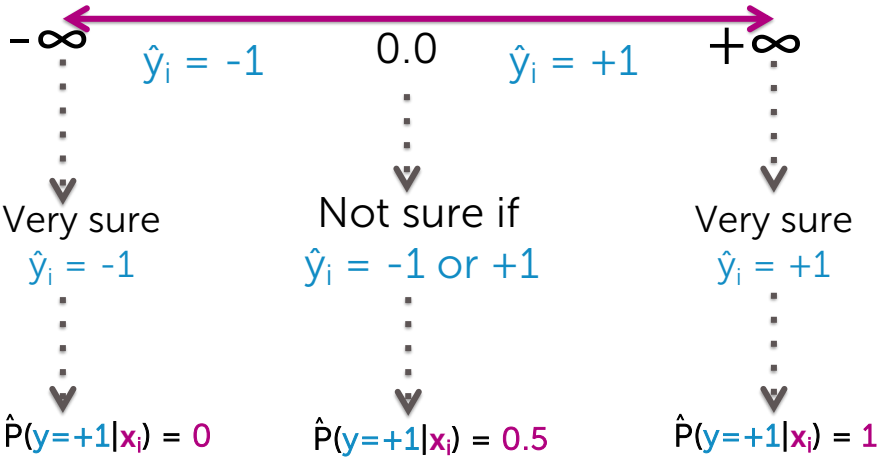


Relate $\text{Score}(\mathbf{x}_i)$ to $\hat{P}(y=+1|\mathbf{x},\hat{\mathbf{w}})$?

21

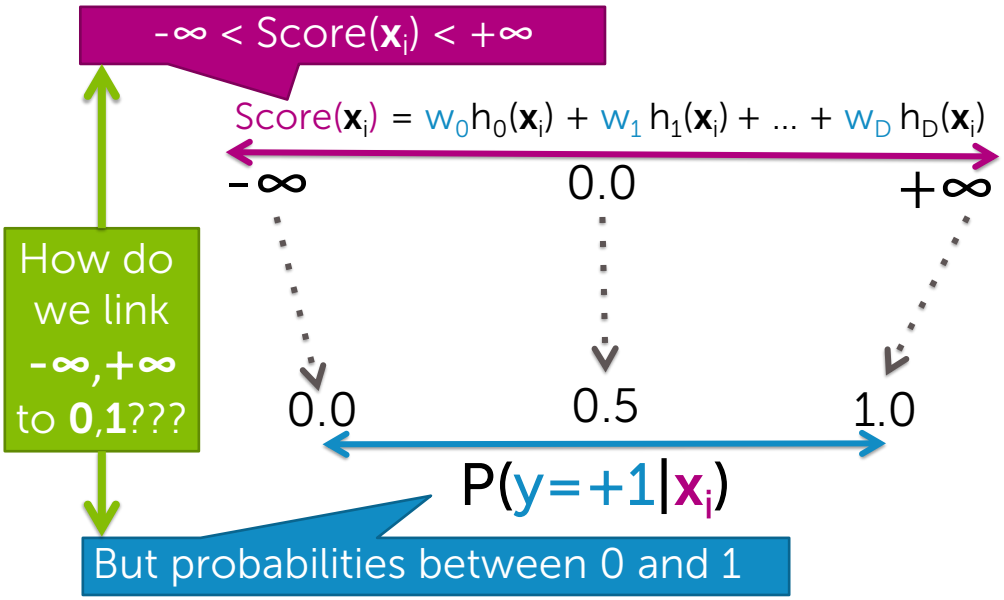
Interpreting $\text{Score}(\mathbf{x}_i)$

$$\text{Score}(\mathbf{x}_i) = \mathbf{w}^T h(\mathbf{x}_i)$$



22

Why not just use regression to build classifier?



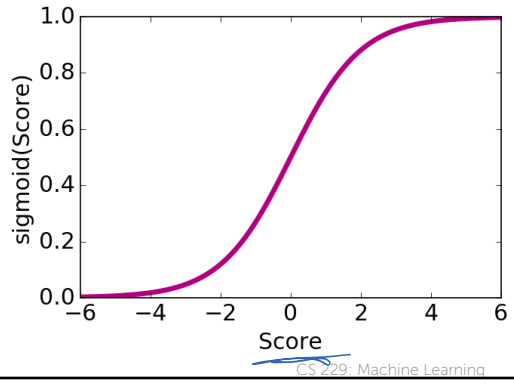
23

Logistic function (sigmoid, logit)

$$\text{sigmoid}(\text{Score}) = \frac{1}{1+e^{-\text{Score}}}$$

"link fcn" used in logistic regression

Score	-∞	-2	0.0	+2	+∞
sigmoid(Score)	0	0.12	0.5	0.88	1

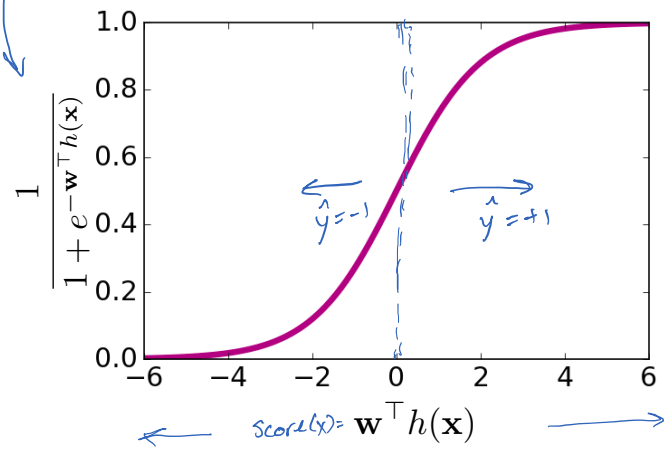


24

Understanding the logistic regression model

$$P(y=+1|\mathbf{x}_i,\mathbf{w}) = \text{sigmoid}(\text{Score}(\mathbf{x}_i)) = \frac{1}{1 + e^{-\mathbf{w}^\top h(\mathbf{x})}}$$

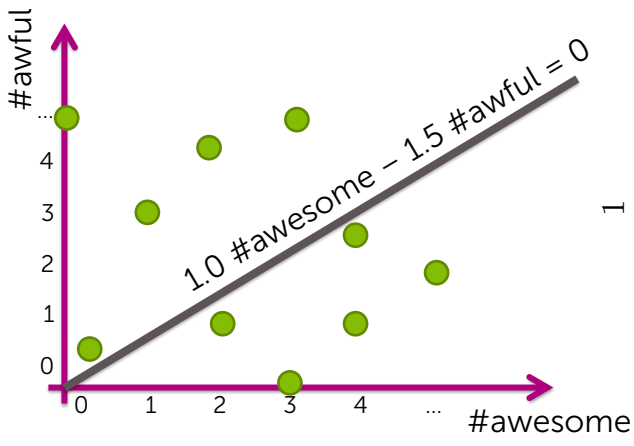
$\mathbf{w}^\top h(\mathbf{x})$ = score(x)

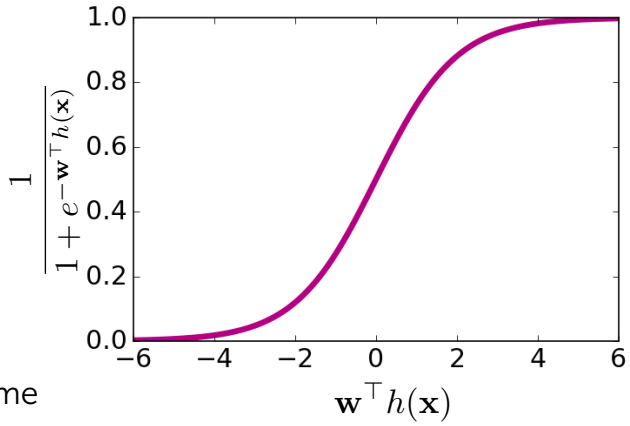


Score($\mathbf{x}_i$)	P(y=+1|$\mathbf{x}_i$,$\mathbf{w}$)
0	0.5
-2	0.12 < 0.5 ⇒ $\hat{y}$ = -1
2	0.88 > 0.5 ⇒ $\hat{y}$ = +1
4	0.98

25

Logistic regression ➔ Linear decision boundary

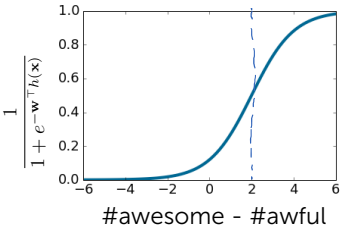




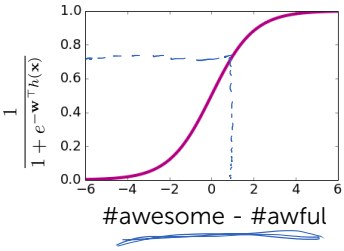
26

Effect of coefficients on logistic regression model

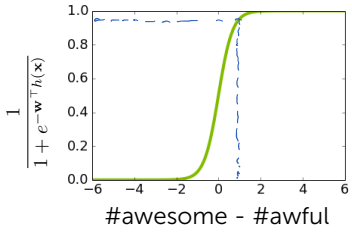
w_0	-2
$w_{\#awesome}$	+1
$w_{\#awful}$	-1



w_0	0
$w_{\#awesome}$	+1
$w_{\#awful}$	-1



w_0	0
$w_{\#awesome}$	+3
$w_{\#awful}$	-3



Score(x) = -2 + #awesome - #awful

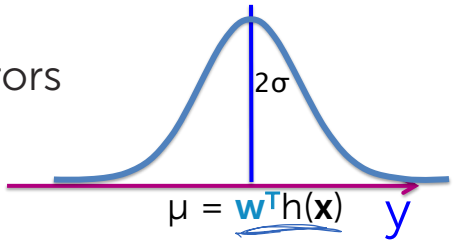
27

Compare and contrast regression models

- Linear regression with Gaussian errors

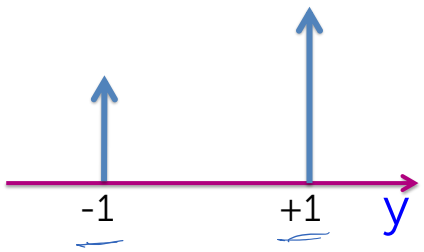
  $y_i = \mathbf{w}^T h(\mathbf{x}_i) + \varepsilon_i \quad \varepsilon_i \sim N(0,\sigma^2)$

  ➔ $p(y|\mathbf{x},\mathbf{w}) = N(y; \mathbf{w}^T h(\mathbf{x}), \sigma^2)$



- Logistic regression

$$P(y|\mathbf{x},\mathbf{w}) = \begin{cases} \dfrac{1}{1+e^{-\mathbf{w}^T h(\mathbf{x})}} & y = +1 \\ \dfrac{e^{-\mathbf{w}^T h(\mathbf{x})}}{1+e^{-\mathbf{w}^T h(\mathbf{x})}} & y = -1 \end{cases}$$



28

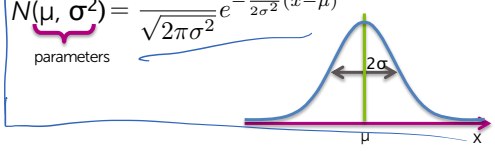
Loss function for logistic regression: (Negative log-) Likelihood for maximum likelihood estimation (MLE)

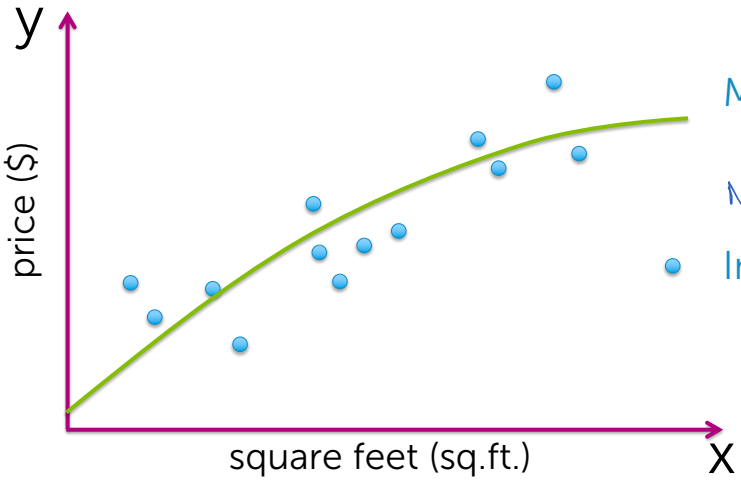
29

Recall: Gaussian linear regression model

$N(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$

parameters





Model for ε_i:

So far: $E[\epsilon_i] = 0$

Now, stronger assumption: $\epsilon_i \sim N(0, \sigma^2)$

Implied distribution on y_i:

$y_i = h^T(x_i) w + \epsilon_i$

"given" const

$y_i | x_i; w \sim N(h^T(x_i) w, \sigma^2)$

30

Recall: Maximum likelihood estimation

$$\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{y} \mid \mathbf{X}; \mathbf{w}) \overset{\epsilon_i \text{ iid}}{=} \arg\max_{\mathbf{w}} \prod_{i=1}^{N} p(y_i \mid x_i; \mathbf{w}) = \arg\max_{\mathbf{w}} \ln \{\cdots\}$$

↑ strictly inc. fcn

Maximize log-likelihood wrt **w**

$$\arg\max_{\mathbf{w}} \ln p(\mathbf{y} \mid \mathbf{X}; \mathbf{w}) = \ln \left\{ \left(\frac{1}{\sigma\sqrt{2\pi}} \right)^{N} \prod_{i=1}^{N} e^{\frac{-(y_i - \mathbf{h}(\mathbf{x}_i)^\top \mathbf{w})^2}{2\sigma^2}} \right\}$$

$$= \arg\max_{\mathbf{w}} N \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) - \sum_{i=1}^{N} \frac{(y_i - h^\top(x_i)\mathbf{w})^2}{2\sigma^2}$$

no w dep.

$$= \arg\min_{\mathbf{w}} \sum_{i=1}^{N} (y_i - h^\top(x_i)\mathbf{w})^2 \leftarrow \text{MLE for Gauss model} = \text{min RSS! (LS line)}$$

31

Finding best coefficients

x[1] = #awesome	x[2] = #awful	y = sentiment
2	1	+1
0	2	-1
3	3	-1
4	1	+1
1	1	+1
2	4	-1
0	3	-1
0	1	-1
2	1	+1

32

Finding best coefficients

x[1] = #awesome	x[2] = #awful	y = sentiment
0	2	-1
3	3	-1
2	4	-1
0	3	-1
0	1	-1
2	4	-1
0	3	-1
0	1	-1

x[1] = #awesome	x[2] = #awful	y = sentiment
2	1	+1
4	1	+1
1	1	+1
2	1	+1
1	1	+1
2	1	+1

33

Finding best coefficients

x[1] = #awesome	x[2] = #awful	y = sentiment
0	2	-1
3	3	-1
2	4	-1
0	3	-1
0	1	-1

x[1] = #awesome	x[2] = #awful	y = sentiment
2	1	+1
4	1	+1
1	1	+1
2	1	+1

$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 0.0$

$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 1.0$

Want $\hat{\mathbf{w}}$ that makes

34

Learn logistic regression model with maximum likelihood estimation (MLE)

Data point	x[1]	x[2]	y	Choose w to maximize
$\mathbf{x}_1, y_1$	2	1	+1	P(y=+1|**x**[1]=2, **x**[2]=1,**w**)
$\mathbf{x}_2, y_2$	0	2	-1	P(y=-1|**x**[1]=0, **x**[2]=2,**w**)
$\mathbf{x}_3, y_3$	3	3	-1	P(y=-1|**x**[1]=3, **x**[2]=3,**w**)
$\mathbf{x}_4, y_4$	4	1	+1	P(y=+1|**x**[1]=4, **x**[2]=1,**w**)

$$p(y_i|x_i,w) = \begin{cases} \frac{1}{1+e^{-w^T h(x_i)}} & \text{if } y_i=+1 \\ \frac{e^{-w^T h(x_i)}}{1+e^{-w^T h(x_i)}} & \text{if } y_i=-1 \end{cases}$$

$p(y_i=+1|\ldots) + p(y_i=-1|\ldots) = 1$

$$\ell(\mathbf{w}) = \underbrace{P(y_1|\mathbf{x}_1,\mathbf{w})\; P(y_2|\mathbf{x}_2,\mathbf{w})\; P(y_3|\mathbf{x}_3,\mathbf{w})\; P(y_4|\mathbf{x}_4,\mathbf{w})}_{\prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})}$$

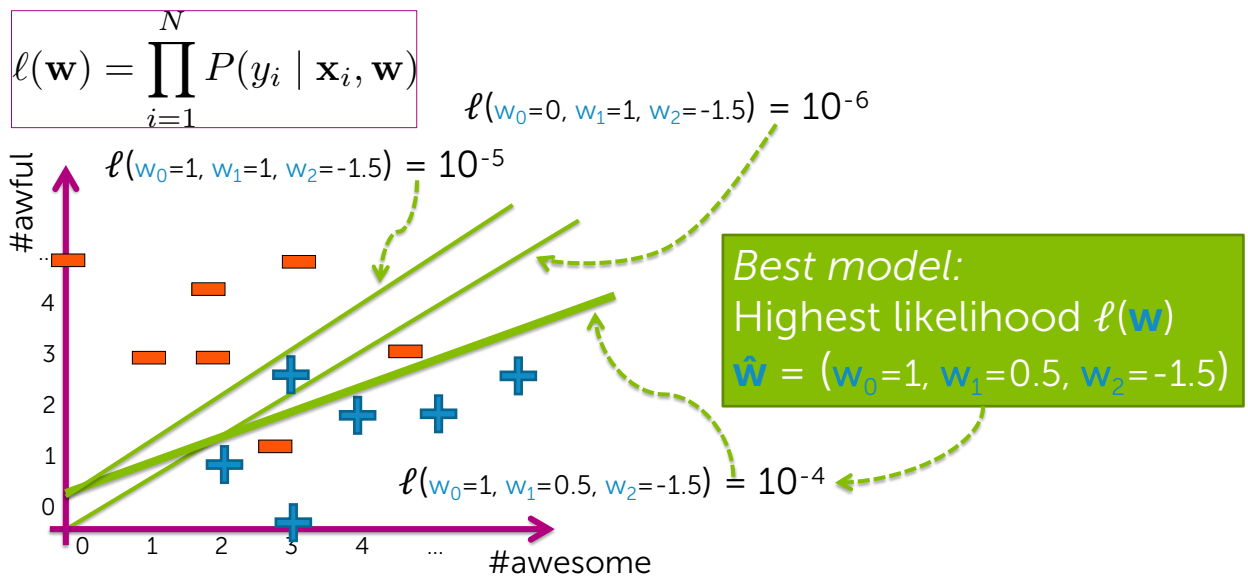
← pick w to make this large

35

Find "best" classifier

Maximize likelihood over all possible w_0, w_1, w_2

$$\ell(\mathbf{w}) = \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})$$



Best model:
Highest likelihood $\ell(\mathbf{w})$
$\hat{\mathbf{w}} = (w_0=1, w_1=0.5, w_2=-1.5)$

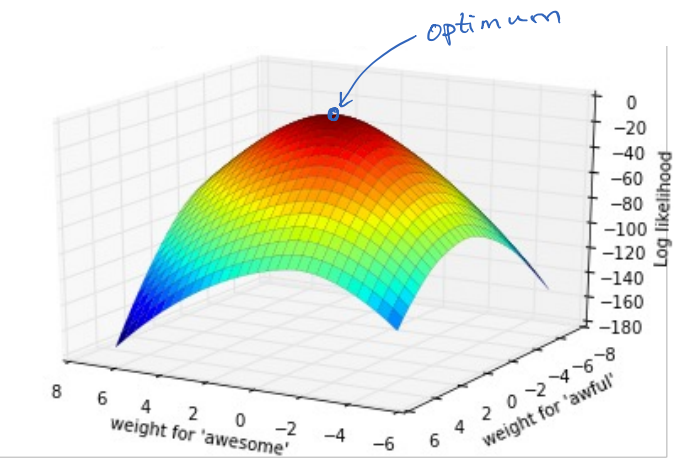
36

If max likelihood as

Gradient ~~descent~~ for logistic regression

37

Maximizing likelihood



Maximize function over all possible w_0, w_1, w_2

$$\max_{w_0,w_1,w_2} \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})$$

$\ell(w_0, w_1, w_2)$ is a function of 3 variables

38

Our optimization objective

- Can compute gradient, but no closed-form solution to:

$$\nabla \ell(\mathbf{w}) = 0$$

- Use gradient descent

- As with MLE for Gaussians, rewrite objective as:

$$\hat{w} = \arg\max_w \ell(w) = \arg\max_w \ell\ell(w)$$

↑ log-likelihood

39

Gradient of logistic log-likelihood

Sum over data points

Feature value

Difference between truth and prediction

$$\frac{\partial \ell\ell(\mathbf{w})}{\partial \mathbf{w}_j} = \sum_{i=1}^{N} h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w})\Big)$$

Δ_i

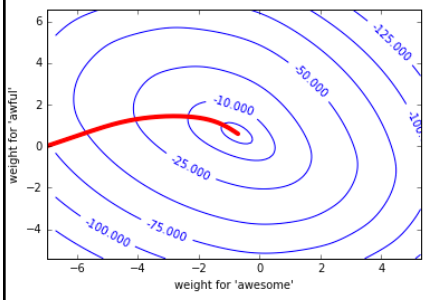
$\mathbb{1}[y_i = +1] = \begin{cases} 1 & \text{if } y_i = +1 \\ 0 & \text{else} \end{cases}$

If $h_j(x_i) = 1$:

	$P(y=+1 \mid x_i, w) \approx 1$	$P(y=+1 \mid x_i, w) \approx 0$
$y_i = +1$	$\Delta_i \approx (1-1) = 0$ → don't change anything	$\Delta_i \approx 1 \Rightarrow$ increase w_j $\Rightarrow P(y=+1 \mid x_i, w)$ increase
$y_i = -1$	$\Delta_i \approx -1 \Rightarrow w_j$ decreases $\Rightarrow P(y=+1 \mid x, w)$ decreases	$\Delta_i \approx 0$ → don't change anything

40

Gradient ascent for logistic regression



init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), t=1

while $||\nabla \ell(\mathbf{w}^{(t)})|| > \varepsilon$

Difference between truth and prediction

for j=0,…,D

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

$w_j^{(t+1)} \leftarrow w_j^{(t)} + \eta$ partial[j]

t ← t + 1

$P(y=+1 \mid \mathbf{x}_i, \mathbf{w}^{(t)}) = \frac{1}{1+e^{-w^{(t)T} h(x_i)}}$

$\frac{\partial \ell\ell(w^{(t)})}{\partial w_j}$

41

Summary for linear classifiers and logistic regression (example of linear classifier)

42

What you can do now…

- Describe decision boundaries and linear classifiers
- Use class probability to express degree of confidence in prediction
- Define a logistic regression model
- Interpret logistic regression outputs as class probabilities
- Describe impact of coefficient values on logistic regression output
- Measure quality of a classifier using the likelihood function
- Learn a logistic regression model with gradient descent for negative log-likelihood loss

43


Linear classifiers: Overfitting

CS 229: Machine Learning
Emily Fox
Stanford University
January 24, 2024

44

Overfitting in classification

45

More complex models tend to have less bias…

Sentiment classifier using single words can do OK, but…

Never classifies correctly:
"The sushi was not good."

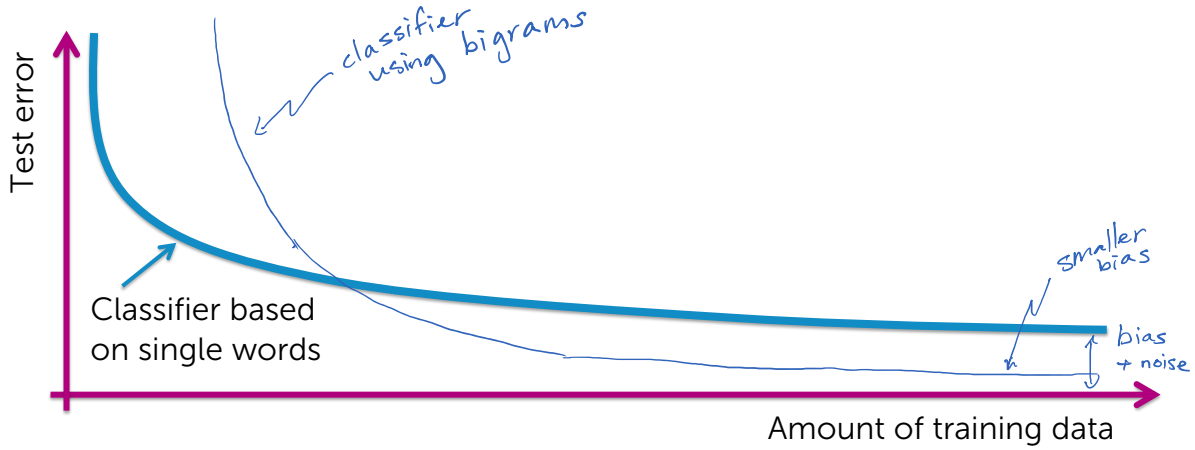
More complex model:
consider pairs of words (bigrams)

Word	Weight
good	+1.5
not good	-2.1

Less bias ➔
potentially more accurate,
needs more data to learn

46

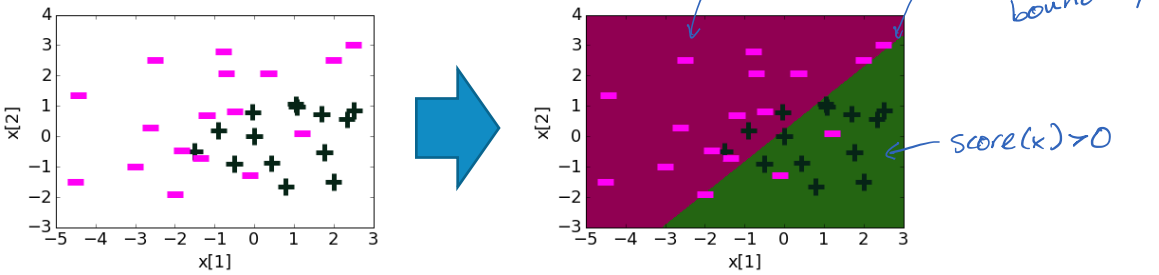
Models with less bias tend to need more data to learn well, but do better with sufficient data



47

Learned decision boundary

Feature	Value	Coefficient learned
$h_0(\mathbf{x})$	1	0.23
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	1.12
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	-1.07

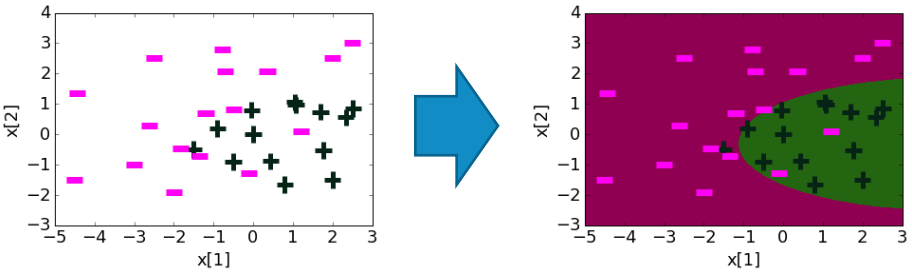


48

Quadratic features (in 2d)

Note: we are not including cross terms for simplicity

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



49

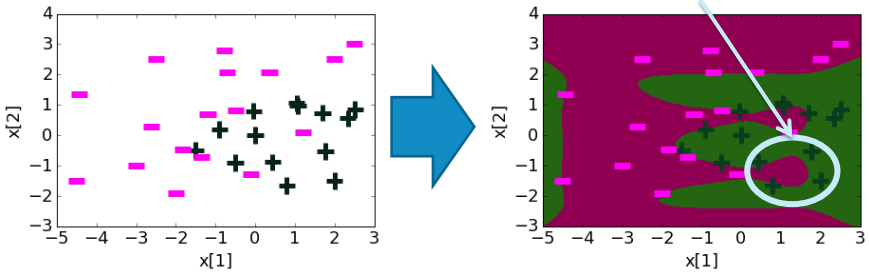
Degree 6 features (in 2d)

Note: we are not including cross terms for simplicity

Feature	Value	Coefficient learned
$h_0(\mathbf{x})$	1	21.6
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	5.3
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	-42.7
$h_3(\mathbf{x})$	$(\mathbf{x}[1])^2$	-15.9
$h_4(\mathbf{x})$	$(\mathbf{x}[2])^2$	-48.6
$h_5(\mathbf{x})$	$(\mathbf{x}[1])^3$	-11.0
$h_6(\mathbf{x})$	$(\mathbf{x}[2])^3$	67.0
$h_7(\mathbf{x})$	$(\mathbf{x}[1])^4$	1.5
$h_8(\mathbf{x})$	$(\mathbf{x}[2])^4$	48.0
$h_9(\mathbf{x})$	$(\mathbf{x}[1])^5$	4.4
$h_{10}(\mathbf{x})$	$(\mathbf{x}[2])^5$	-14.2
$h_{11}(\mathbf{x})$	$(\mathbf{x}[1])^6$	0.8
$h_{12}(\mathbf{x})$	$(\mathbf{x}[2])^6$	-8.6

Coefficient values getting large

Score($\mathbf{x}$) < 0



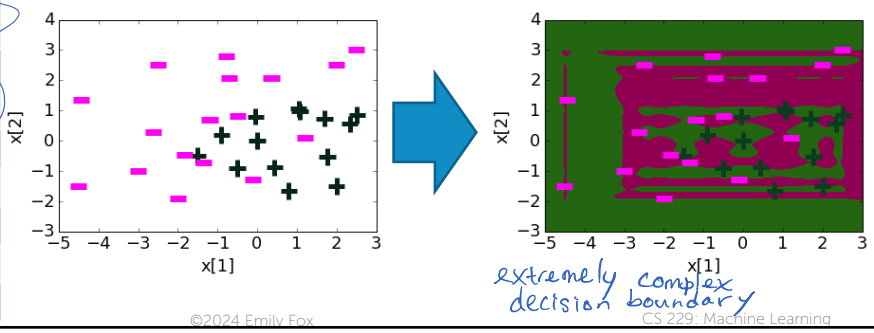
50

Degree 20 features (in 2d)

Note: we are not including cross terms for simplicity

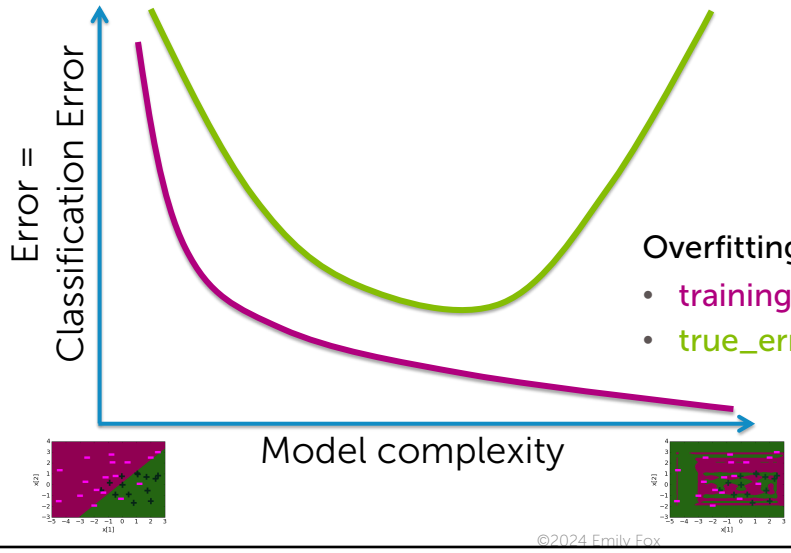
Feature	Value	Coefficient learned
$h_0(\mathbf{x})$	1	8.7
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	5.1
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	78.7
...	...	...
$h_{11}(\mathbf{x})$	$(\mathbf{x}[1])^6$	-7.5
$h_{12}(\mathbf{x})$	$(\mathbf{x}[2])^6$	3803
$h_{13}(\mathbf{x})$	$(\mathbf{x}[1])^7$	21.1
$h_{14}(\mathbf{x})$	$(\mathbf{x}[2])^7$	-2406
...	...	...
$h_{37}(\mathbf{x})$	$(\mathbf{x}[1])^{19}$	$-2*10^{-6}$
$h_{38}(\mathbf{x})$	$(\mathbf{x}[2])^{19}$	-0.15
$h_{39}(\mathbf{x})$	$(\mathbf{x}[1])^{20}$	$-2*10^{-8}$
$h_{40}(\mathbf{x})$	$(\mathbf{x}[2])^{20}$	0.03

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



51

Overfitting in classification



Overfitting if there exists w^*:

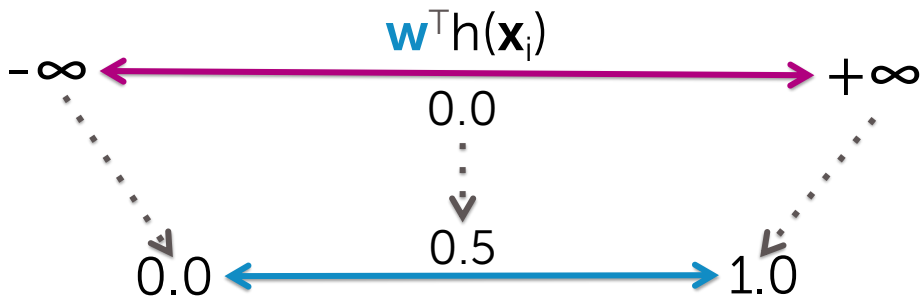
- training_error(w^*) > training_error($\hat{w}$)
- true_error(w^*) < true_error($\hat{w}$)

52

Overfitting in classifiers ➔
Overconfident predictions

53

Logistic regression model



$$P(y=+1|\mathbf{x}_i,\mathbf{w}) = \text{sigmoid}(\mathbf{w}^T h(\mathbf{x}_i))$$

54

The subtle (negative) consequence of overfitting in logistic regression

Overfitting ➔ Large coefficient values

$\hat{\mathbf{w}}^T h(\mathbf{x}_i)$ is very positive (or very negative) ➔ **sigmoid**($\hat{\mathbf{w}}^T h(\mathbf{x}_i)$) goes to 1 (or to 0)

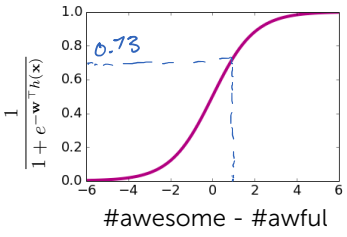
Model becomes extremely overconfident of predictions

55

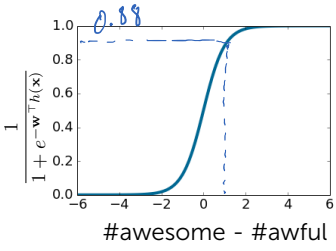
Effect of coefficients on logistic regression model

Input **x**: #awesome=2, #awful=1

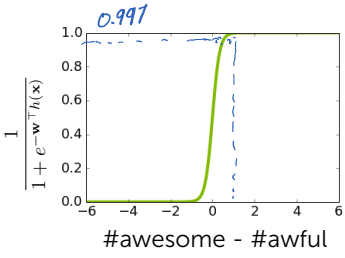
w_0	0
$w_{\#awesome}$	+1
$w_{\#awful}$	-1



w_0	0
$w_{\#awesome}$	+2
$w_{\#awful}$	-2



w_0	0
$w_{\#awesome}$	+6
$w_{\#awful}$	-6

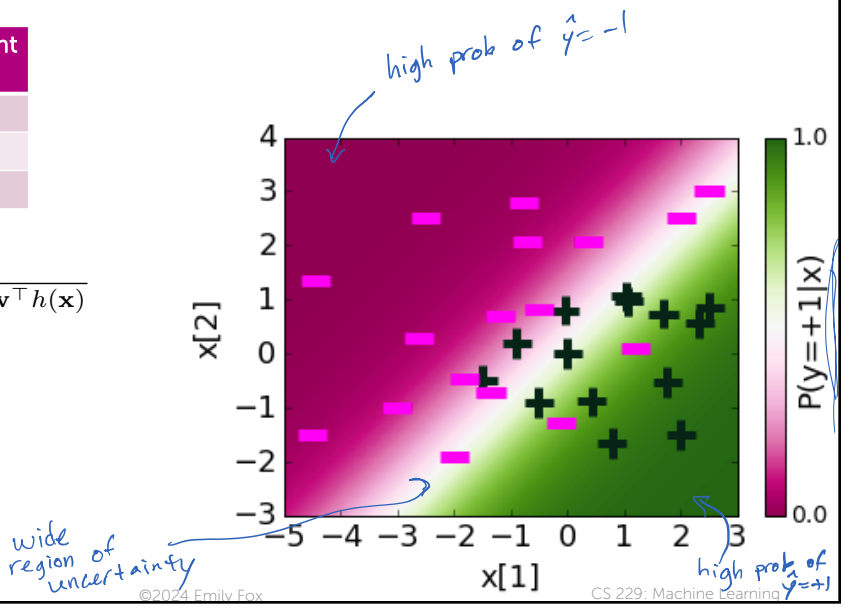


56

Learned probabilities

Feature	Value	Coefficient learned
$h_0(\mathbf{x})$	1	0.23
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	1.12
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	-1.07

$$P(y = +1 \mid \mathbf{x}, \mathbf{w}) = \frac{1}{1 + e^{-\mathbf{w}^\top h(\mathbf{x})}}$$

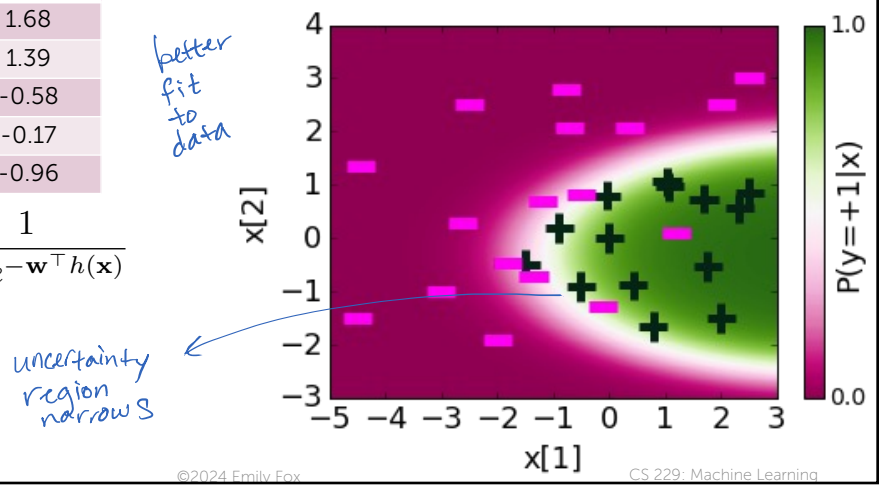


57

Quadratic features: Learned probabilities

Feature	Value	Coefficient learned
$h_0(\mathbf{x})$	1	1.68
$h_1(\mathbf{x})$	$\mathbf{x}[1]$	1.39
$h_2(\mathbf{x})$	$\mathbf{x}[2]$	-0.58
$h_3(\mathbf{x})$	$(\mathbf{x}[1])^2$	-0.17
$h_4(\mathbf{x})$	$(\mathbf{x}[2])^2$	-0.96

$$P(y = +1 \mid \mathbf{x}, \mathbf{w}) = \frac{1}{1 + e^{-\mathbf{w}^\top h(\mathbf{x})}}$$

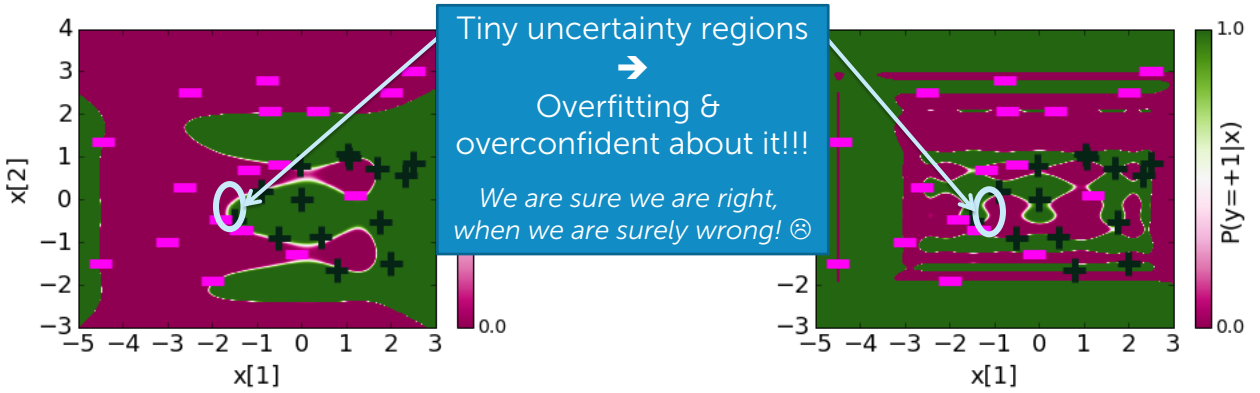


58

Overfitting ➔ Overconfident predictions

Degree 6: Learned probabilities

Degree 20: Learned probabilities

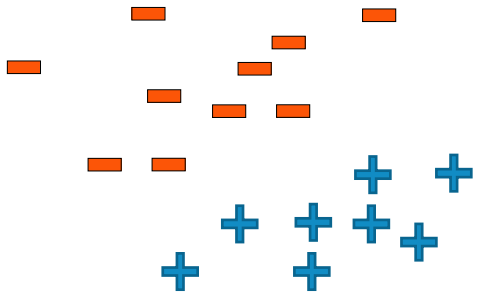


59

Overfitting in logistic regression: Another perspective

60

Linearly-separable data



Note 1: If you are using D features, linear separability happens in a D-dimensional space

Note 2: If you have enough features, data are (almost) always linearly separable

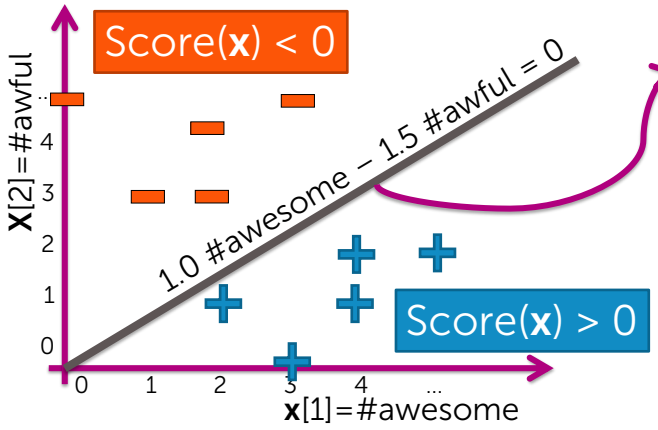
Data are linearly separable if:

- There exist coefficients $\hat{\mathbf{w}}$ such that:
  - For all positive training data
    $score(x) = \hat{w}^T h(x) > 0$
  - For all negative training data
    $score(x) = \hat{w}^T h(x) < 0$

$\text{training\_error}(\hat{\mathbf{w}}) = 0$

61

Effect of linear separability on coefficients



same decision boundaries

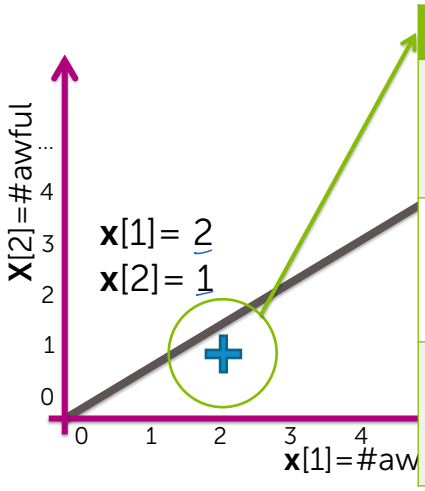
Data are linearly separable with $\hat{w}_1$=1.0 and $\hat{w}_2$=-1.5

Data also linearly separable with $\hat{w}_1$=10 and $\hat{w}_2$=-15

Data also linearly separable with $\hat{w}_1$=10^9 and $\hat{w}_2$=-1.5x10^9

62

Maximum likelihood estimation (MLE)
prefers most certain model ➔
Coefficients go to infinity for linearly-separable data!!!



$P(y=+1|x_i, \hat{w})$

$\frac{1}{1+e^{-(2 \cdot 1.5)}} = \frac{1}{1+e^{-0.5}} = 0.62$

$\frac{1}{1+e^{-5}} = 0.99$

$\frac{1}{1+e^{-0.5 \times 10^9}} \approx 1$

Data is linearly separable with $\hat{w}_1$=1.0 and $\hat{w}_2$=-1.5

Data also linearly separable with $\hat{w}_1$=10 and $\hat{w}_2$=-15

Data also linearly separable with $\hat{w}_1$=10^9 and $\hat{w}_2$=-1.5x10^9

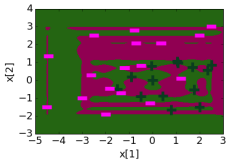
63

Overfitting in logistic regression is "twice as bad"

Learning tries to find decision boundary that separates data

If data are linearly separable

Overly complex boundary



Coefficients go to infinity!

$\hat{w}_1$=10^9
$\hat{w}_2$=-1.5x10^9

64

Penalizing large coefficients to mitigate overfitting

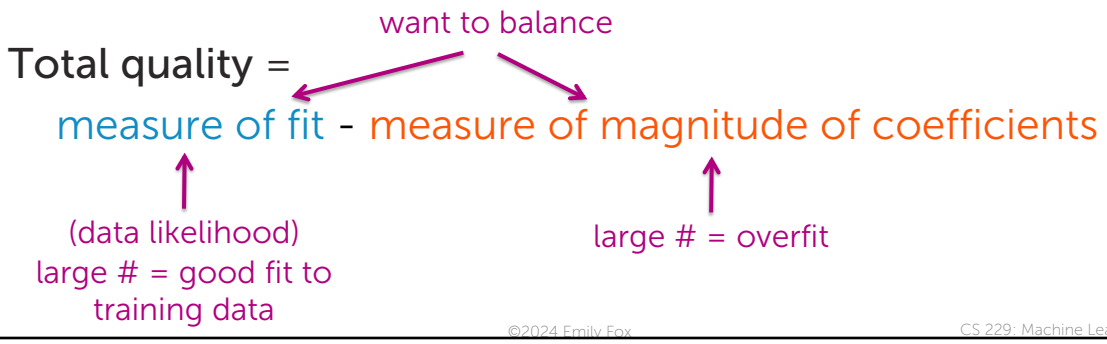
65

Desired total cost format

Want to balance:

i. How well function fits data

ii. Magnitude of coefficients



66

L_2 regularized logistic regression

67

Consider resulting objective

Select $\hat{\mathbf{w}}$ to maximize:

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

tuning parameter = balance of fit and magnitude

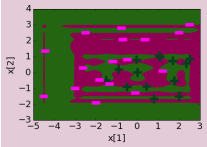
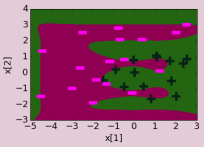
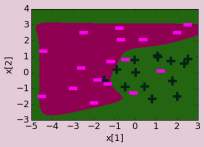
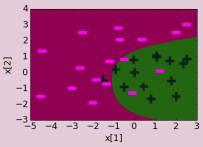
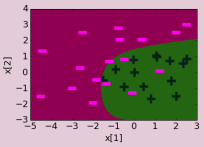
L_2 regularized logistic regression

Pick λ using:
- Validation set (for large datasets)
- Cross-validation (for smaller datasets)

(as in ridge/lasso regression)

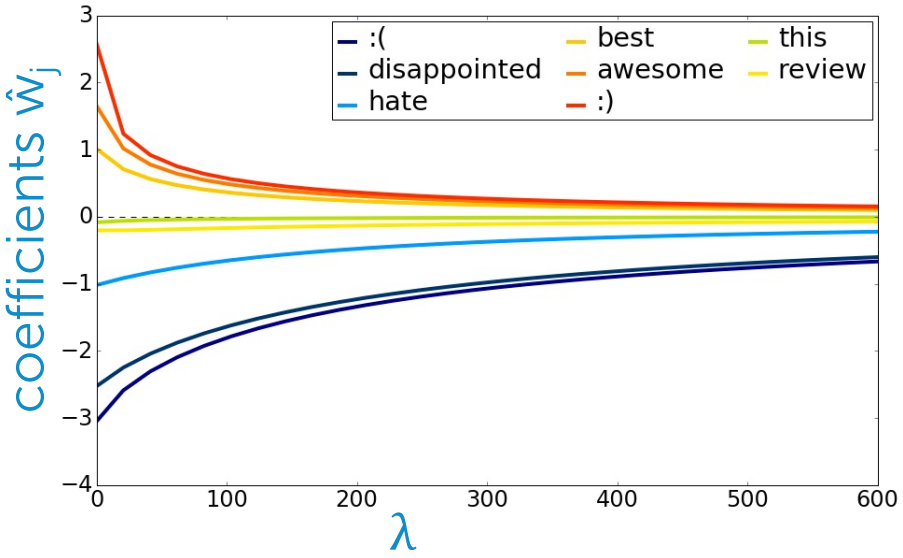
68

Degree 20 features, effect of regularization penalty λ

Regularization	λ = 0	λ = 0.00001	λ = 0.001	λ = 1	λ = 10
Range of coefficients	-3170 to 3803	-8.04 to 12.14	-0.70 to 1.25	-0.13 to 0.57	-0.05 to 0.22
Decision boundary					

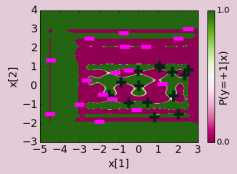
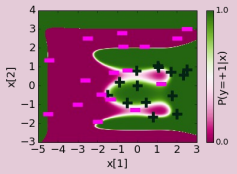
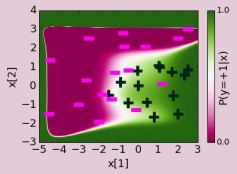
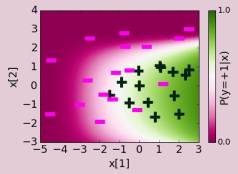
69

Coefficient path

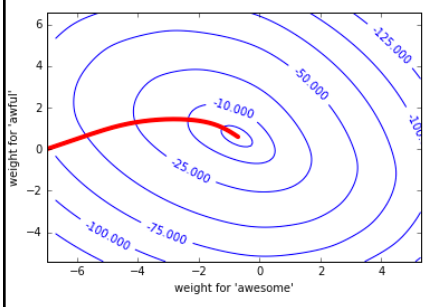


70

Degree 20 features: regularization reduces "overconfidence"

Regularization	λ = 0	λ = 0.00001	λ = 0.001	λ = 1
Range of coefficients	-3170 to 3803	-8.04 to 12.14	-0.70 to 1.25	-0.13 to 0.57
Learned probabilities				

Gradient ascent for L_2 regularized logistic regression



init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), t=1

while not converged:

for j=0,…,D

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

$\mathbf{w}_j^{(t+1)} \leftarrow \mathbf{w}_j^{(t)} + \eta\,(\text{partial[j]} - 2\lambda\,\mathbf{w}_j^{(t)})$

Only change

$t \leftarrow t + 1$

L_1 regularized logistic regression

73

Sparse logistic regression with L_1 penalty

Select $\hat{\mathbf{w}}$ to maximize:

$$\ell(\mathbf{w}) - \lambda||\mathbf{w}||_1$$

tuning parameter = balance of fit and magnitude

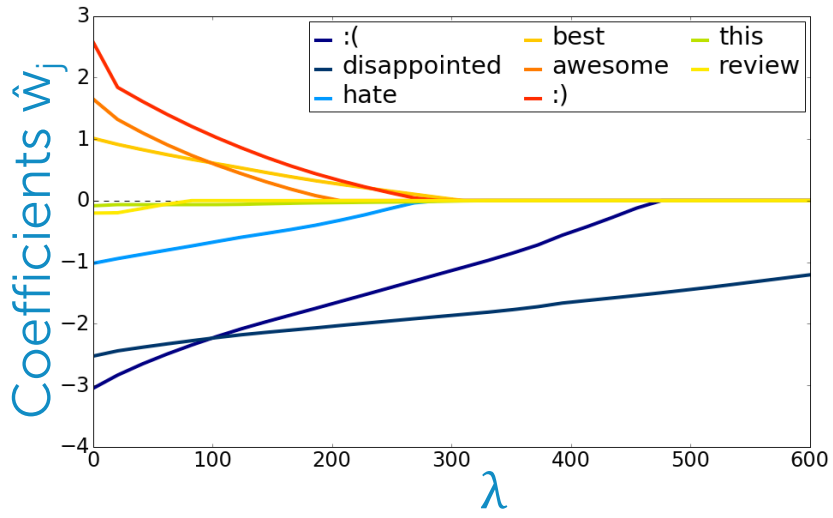
L_1 regularized logistic regression

Pick λ using:
- Validation set (for large datasets)
- Cross-validation (for smaller datasets)

(as in ridge/lasso regression)

74

Coefficient path – L_1 penalty



75

Summary of overfitting in logistic regression

76

What you can do now…

- Describe symptoms and effects of overfitting in classification
  - Identify when overfitting is happening
  - Relate large learned coefficients to overfitting
  - Describe the impact of overfitting on decision boundaries and predicted probabilities of linear classifiers
- Use regularization to mitigate overfitting
  - Motivate the form of L2 regularized logistic regression quality metric
  - Describe the use of L1 regularization to obtain sparse logistic regression solutions
  - Describe what happens to estimated coefficients as tuning parameter λ is varied
  - Estimate L2 regularized logistic regression coefficients using gradient ascent
  - Interpret coefficient path plot

77


Linear classifiers: More fundamentals

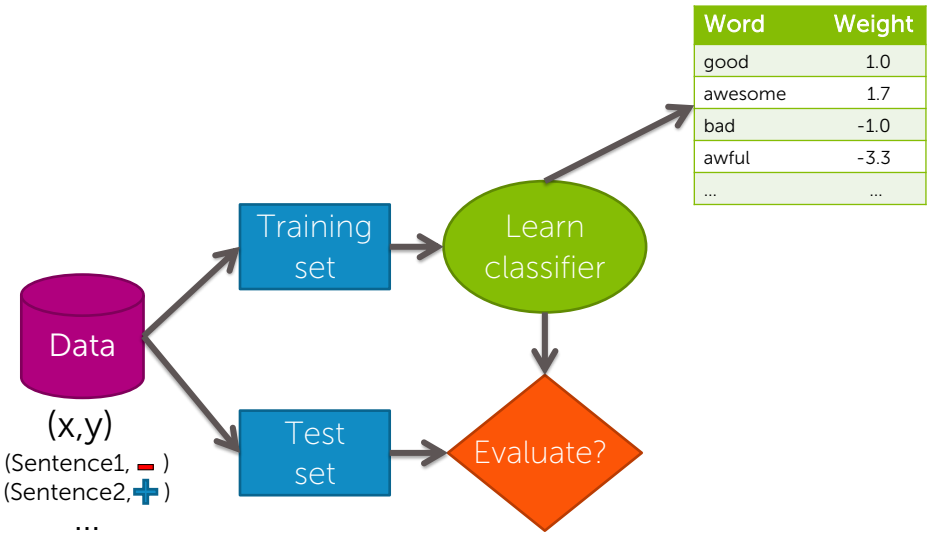
CS 229: Machine Learning
Emily Fox
Stanford University
January 24, 2024

78

Evaluating a classifier

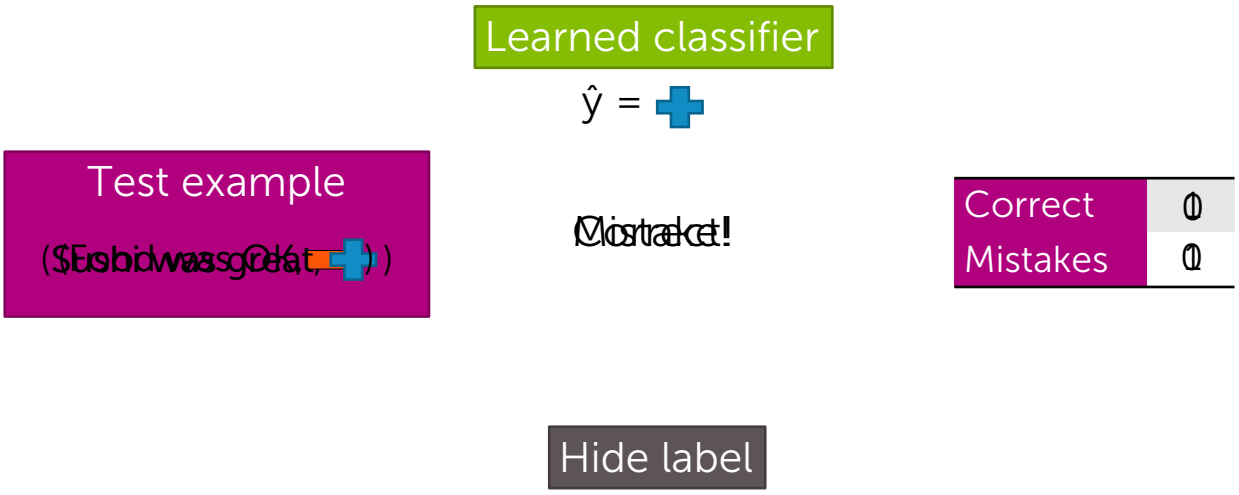
79

Training a classifier = Learning the weights
Then evaluate…



80

Classification error



81

Classification error & accuracy

- Error measures fraction of mistakes

$$\text{error} = \frac{\#\text{ of mistakes}}{\text{total }\#\text{ of sentences}}$$

test set

  - Best possible value is 0.0

error = 1 − accuracy

- Often, measure **accuracy**
  - Fraction of correct predictions

$$\text{accuracy} = \frac{\#\text{ correct}}{\text{total }\#\text{ of sentences}}$$

  - Best possible value is 1.0

82

What's a good accuracy?

83

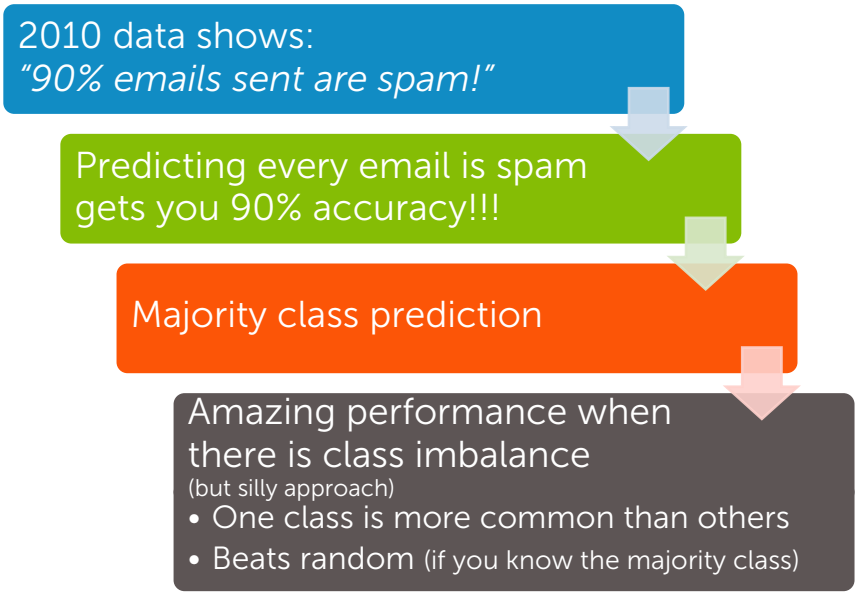
What if you ignore the sentence, and just guess?

- For binary classification:
  - Half the time, you'll get it right! (on average)
    - ➔ accuracy = 0.5
- For k classes, accuracy = 1/k
  - 1/3 for 3 classes, 1/4 for 4 classes,...

At the very, very, very least,
you should healthily beat random...
Otherwise, it's (usually) pointless...

84

Is a classifier with 90% accuracy good? Depends…



85

So, always be digging in and asking the hard questions about reported accuracies

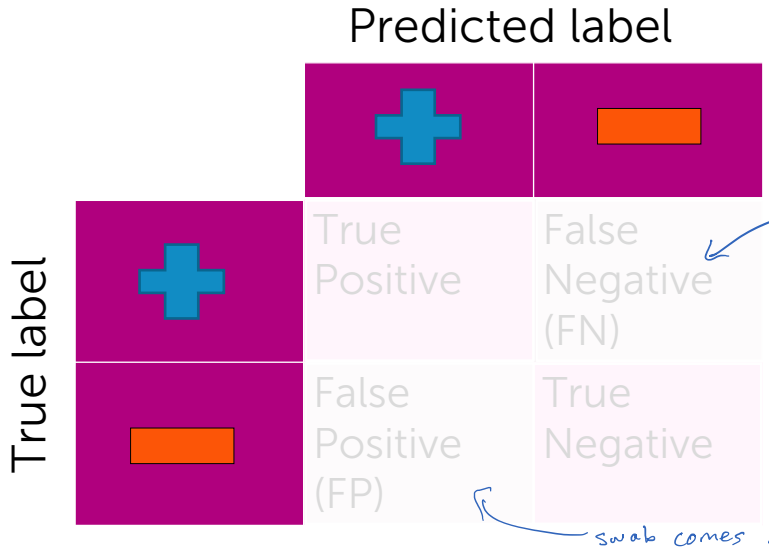
- Is there class imbalance?
- How does it compare to a simple, baseline approach?
  - Random guessing
  - Majority class
  - …
- Most importantly:
  What accuracy does my application need?
  - What is good enough for my user's experience?
  - What is the impact of the mistakes we make?

86

False positives, false negatives, and confusion matrices

87

Types of mistakes



88

Cost of different types of mistakes can be different (& high) in some applications

	Spam filtering	Medical diagnosis
False negative	Annoying	Disease not treated
False positive	Email lost	Wasteful treatment

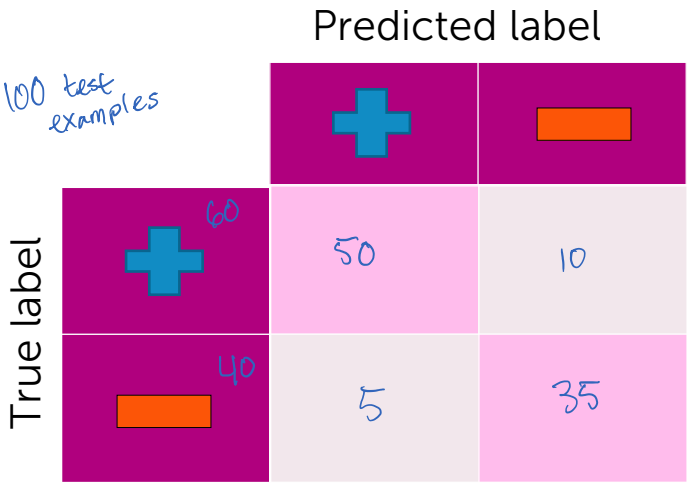
depends

higher cost

anxiety...

89

Confusion matrix – binary classification



$$\text{accuracy} = \frac{50+35}{100} = 0.85$$

90

Encoding categorical inputs

OPTIONAL

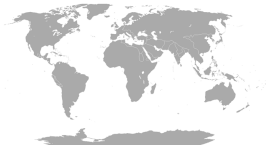
91

Categorical inputs

- Numeric inputs:
  - #awesome, age, salary,…
  - Intuitive when multiplied by coefficient
    - e.g., 1.5 #awesome
- Categorical inputs:

Numeric value, but should be interpreted as category
(94305 not about 9x larger than 10005)

Gender
(Male, Female,…)



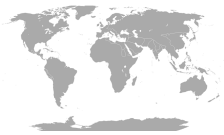
Country of birth
(Argentina, Brazil, USA,…)

Zipcode
(10005, 94305,…)

How do we multiply category by coefficient???
Must convert categorical inputs into numeric features

92

Encoding categories as numeric features

x = 

Country of birth
(Argentina, Brazil, USA,...)

196 categories

1-hot encoding

x	$h_1(x)$	$h_2(x)$	...	$h_{195}(x)$	$h_{196}(x)$
Brazil	0	1	0	0	0
Zimbabwe	0	0	0	0	1

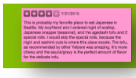
196 features

x = 

Restaurant review
(Text data)

10,000 words in vocabulary

Bag of words

x	$h_1(x)$	$h_2(x)$	...	$h_{9999}(x)$	$h_{10000}(x)$
	2	0	0	0	3

10,000 features

93

Multiclass classification using 1 versus all

OPTIONAL

94

Multiclass classification

Input: **x**
Image pixels

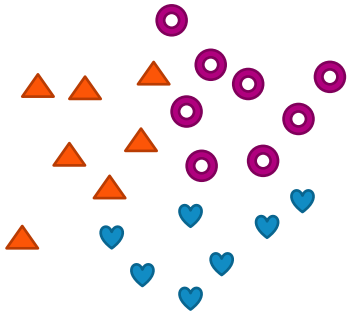
Output: y
Object in image

95

Multiclass classification formulation

- C possible classes:
  - y can be 1, 2,…, C
- N datapoints:

before $y \in \{-1, 1\}$



Data point	x[1]	x[2]	y
$\mathbf{x}_1, y_1$	2	1	▲
$\mathbf{x}_2, y_2$	0	2	♥
$\mathbf{x}_3, y_3$	3	3	○
$\mathbf{x}_4, y_4$	4	1	○

Learn:

$\hat{P}(y=▲ \mid x)$

$\hat{P}(y=♥ \mid x)$

$\hat{P}(y=○ \mid x)$

96

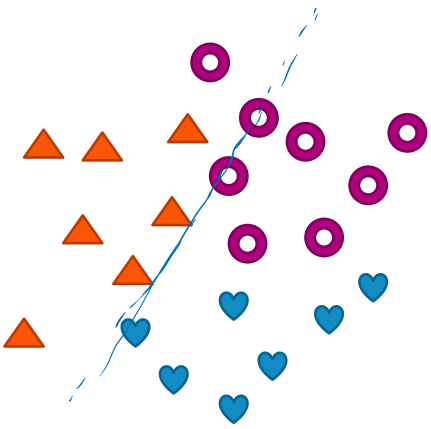
1 versus all:

Estimate $\hat{P}(y=\blacktriangle \mid x)$ using 2-class model

+1 class: points with $y_i=$ ▲
-1 class: points with $y_i=$ ♥ OR ○

Train classifier: $\hat{P}_{\blacktriangle}(y=+1|x)$

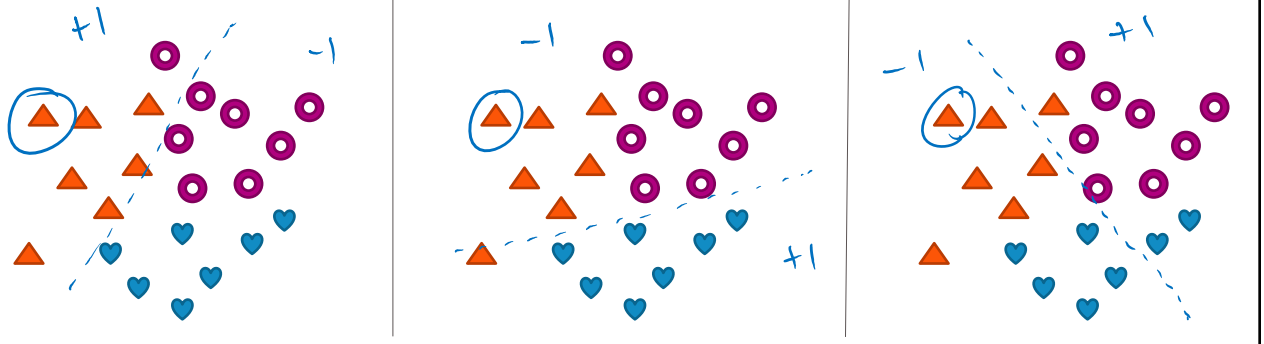
Predict: $\hat{P}(y=\blacktriangle \mid x_i) = \hat{P}_{\blacktriangle}(y=+1|x_i)$



97

1 versus all: simple multiclass classification using *C* 2-class models

$\hat{P}(y=\blacktriangle \mid x_i) = \hat{P}_{\blacktriangle}(y=+1|x_i, w_{\blacktriangle})$ $\hat{P}(y=\heartsuit \mid x_i) = \hat{P}_{\heartsuit}(y=+1|x_i, w_{\heartsuit})$ $\hat{P}(y=\bigcirc \mid x_i) = \hat{P}_{\bigcirc}(y=+1|x_i, w_{\bigcirc})$



98

Input: $\mathbf{x}_i$

Multiclass training

$\hat{P}_c(y=+1|\mathbf{x})$ = estimate of 1 vs all model for each class

Predict most likely class

max_prob = 0; $\hat{y}$ = 0
For c = 1,…,C:
  If $\hat{P}_c(y=+1|\mathbf{x}_i)$ > max_prob:
    $\hat{y}$ = c
    max_prob = $\hat{P}_c(y=+1|\mathbf{x}_i)$

Logistic Regression: Gradient descent details

CS 229: Machine Learning
Emily Fox
Stanford University
January 24, 2024

OPTIONAL

Step 1: Rewrite log-likelihood

For simpler math, we'll rewrite likelihood with indicators:

$$\ell\ell(\mathbf{w}) = \sum_{i=1}^{N} \ln P(y_i \mid \mathbf{x}_i, \mathbf{w})$$

$$= \sum_{i=1}^{N} [\mathbb{1}[y_i = +1] \ln P(y = +1 \mid \mathbf{x}_i, \mathbf{w}) + \mathbb{1}[y_i = -1] \ln P(y = -1 \mid \mathbf{x}_i, \mathbf{w})]$$

Indicator function:

$$\mathbb{1}[y_i = +1] = \begin{cases} 1 & \text{if } y_i = +1 \\ 0 & \text{if } y_i = -1 \end{cases}$$

101

Step 2: Express probabilities in terms of **w** and h(**x**)

Probability model predicts y=+1:

$$P(y=+1|\mathbf{x},\mathbf{w}) = \frac{1}{1 + e^{-\mathbf{w}^T h(\mathbf{x})}}$$

Probability model predicts y=-1:

$$P(y=-1|x,w) = 1 - P(y=+1|x,w)$$

$$= \frac{e^{-w^T h(x)}}{1 + e^{-w^T h(x)}}$$

102

Step 3: Plugging in for 1 data point

$$P(y=+1 \mid \mathbf{x}, \mathbf{w}) = \frac{1}{1+e^{-\mathbf{w}^\top h(\mathbf{x})}} \qquad P(y=-1 \mid \mathbf{x}, \mathbf{w}) = \frac{e^{-\mathbf{w}^\top h(\mathbf{x})}}{1+e^{-\mathbf{w}^\top h(\mathbf{x})}}$$

$$\ell\ell(\mathbf{w}) = \mathbb{1}[y_i=+1] \ln P(y=+1 \mid \mathbf{x}_i, \mathbf{w}) + \mathbb{1}[y_i=-1] \ln P(y=-1 \mid \mathbf{x}_i, \mathbf{w})$$

ln num − ln den

$$\ell\ell(w) = -\mathbb{1}[y_i=+1] \ln\left(1+e^{-w^\top h(x)}\right) + \left(1-\mathbb{1}[y_i=+1]\right)\left(-w^\top h(x) - \ln\left(1+e^{-w^\top h(x)}\right)\right)$$

$$= \left(1-\mathbb{1}[y_i=+1]\right) w^\top h(x_i) - \ln\left(1+e^{-w^\top h(x_i)}\right)$$

Step 4: Gradient for 1 data point

$$\ell\ell(\mathbf{w}) = -(1-\mathbb{1}[y_i=+1])\mathbf{w}^\top h(\mathbf{x}_i) - \ln\left(1+e^{-\mathbf{w}^\top h(\mathbf{x}_i)}\right)$$

$$\frac{\partial \ell\ell(w)}{\partial w_j} = -\left(1-\mathbb{1}[y_i=+1]\right) h_j(x_i)$$

$$+ h_j(x_i)\left(1-P(y=+1 \mid x_i, w)\right)$$

$$= h_j(x_i)\left[\mathbb{1}[y_i=+1] - P(y=+1 \mid x_i, w)\right]$$

① $\frac{\partial}{\partial w_j} w^\top h(x_i) = h_j(x_i)$

② $\frac{\partial}{\partial w_j} \ln\left(1+e^{-w^\top h(x_i)}\right)$

$$= \frac{1}{1+e^{-w^\top h(x_i)}} e^{-w^\top h(x_i)} \left(-h_j(x_i)\right)$$

$$= -h_j(x_i) \frac{e^{-w^\top h(x_i)}}{1+e^{-w^\top h(x_i)}}$$

Step 5: Gradient over all data points

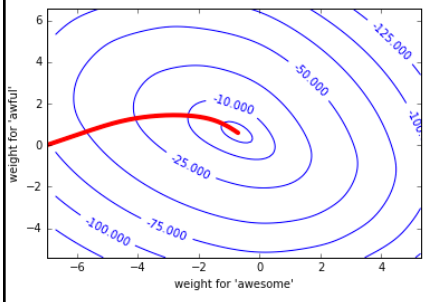
Sum over data points

Feature value

Difference between truth and prediction

$$\frac{\partial \ell\ell(\mathbf{w})}{\partial \mathbf{w}_j} = \sum_{i=1}^{N} h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w})\Big)$$

Gradient ascent for logistic regression



init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), t=1

while $||\nabla \ell(\mathbf{w}^{(t)})|| > \varepsilon$

Difference between truth and prediction

for j=0,…,D

partial[j] = $\sum_{i=1}^{N} h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w}^{(t)})\Big)$

$w_j^{(t+1)} \leftarrow w_j^{(t)} + \eta$ partial[j]

t ← t + 1

Choosing the step size η

107

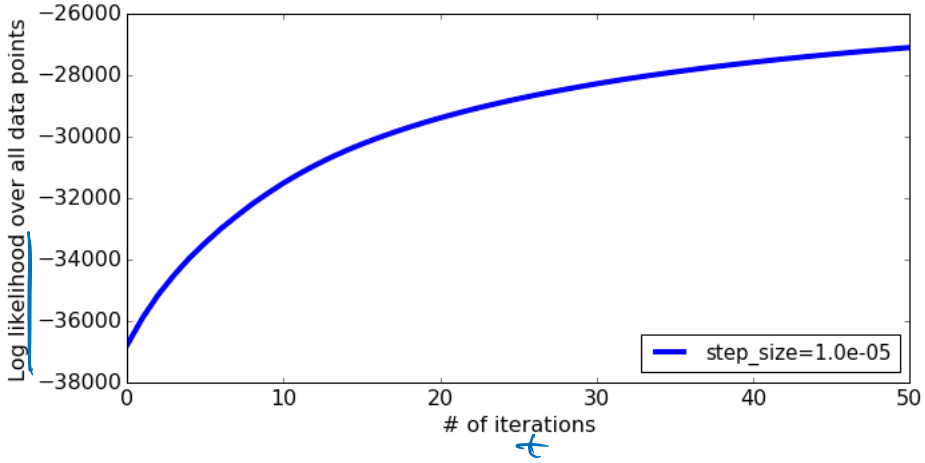
Simple rule of thumb for picking step size η

- Unfortunately, picking step size requires a lot of trial & error ☹
- Try a several values, exponentially spaced
  - **Goal:** plot learning curves to
    - find one η that is too small (smooth but moving too slowly)
    - find one η that is too large (oscillation or divergence)
- Try values in between to find "best" η

- *Advanced tip:* can also try step size that decreases with iterations, e.g., $\eta_t = \frac{\eta_0}{t}$

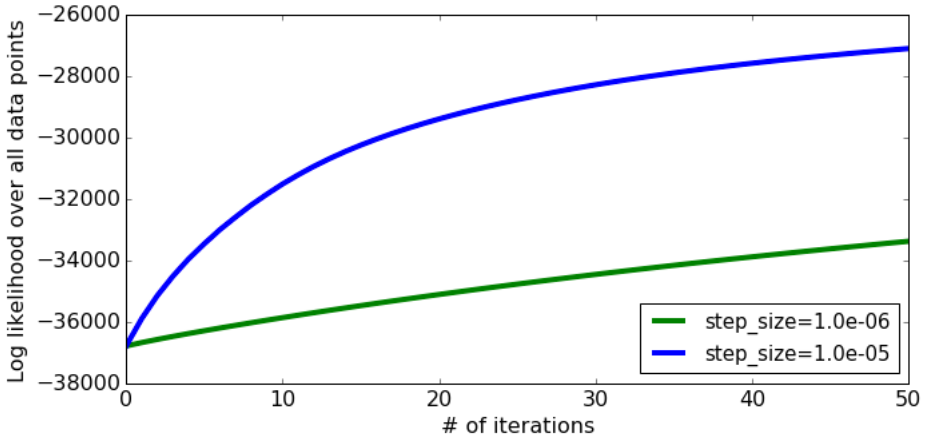
108

Learning curve:
Plot quality (likelihood) over iterations



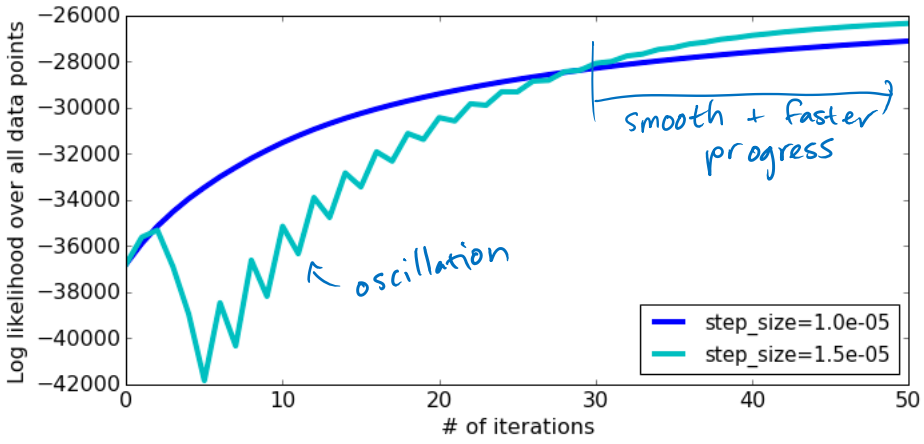
109

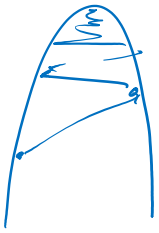
If step size is too small, can take a long time to converge



110

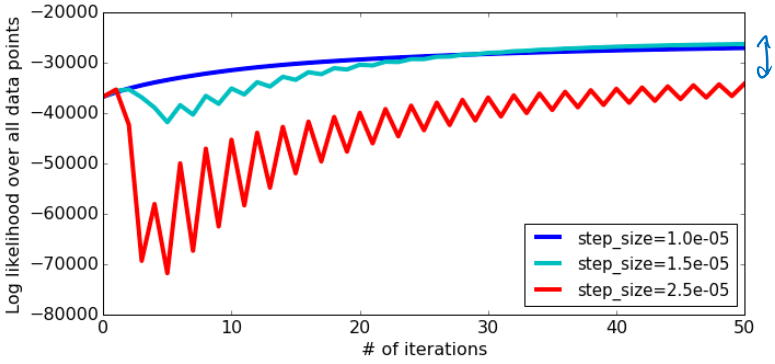
Compare converge with different step sizes



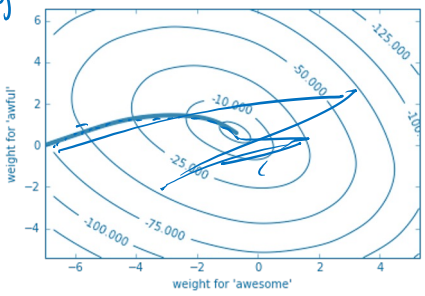


111

Careful with step sizes that are too large

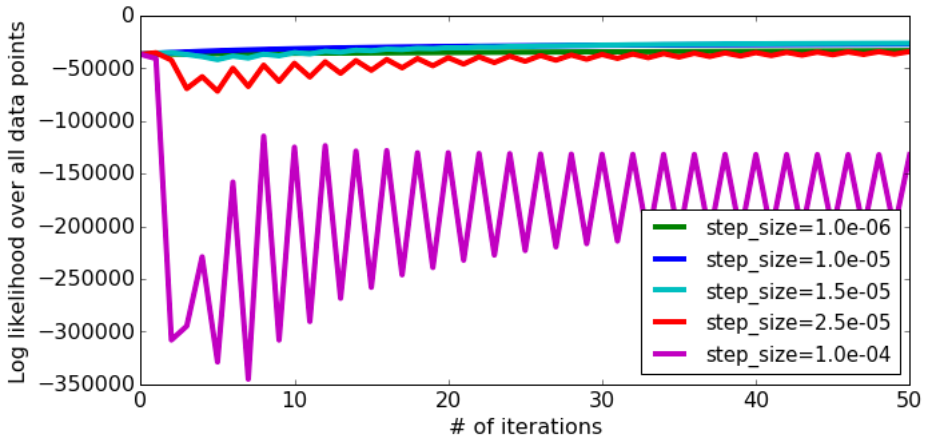


J gap in l(w)



112

Very large step sizes can even cause divergence or wild oscillations



113