Phonological Variation Seminar (Smith, Spring 2017)

Today

1. MaxEnt and logistic regression models

MaxEnt and logistic regression models

1. Roadmap
    a. Linear regression
    b. Odds, log-odds, and the logit function
    c. Logistic regression = MaxEnt, with slight differences in implementation

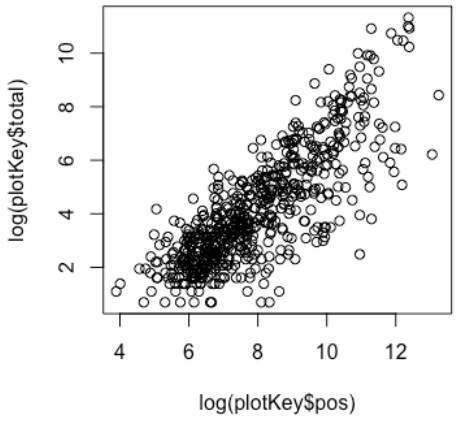
2. Linear regression: a method for modeling the relationship between a dependent variable (*y*) and one or more explanatory variables (X)

3. A **simple linear regression** models *y* given *X* where there's only one explanatory variable. Examples:
    a. Predict height given weight (height ~ weight)
    b. Predict vowel length given speech rate (length ~ rate)
    c. For adjectives, predict frequency of comparative forms given frequency of non-comparative forms (=positive frequency)

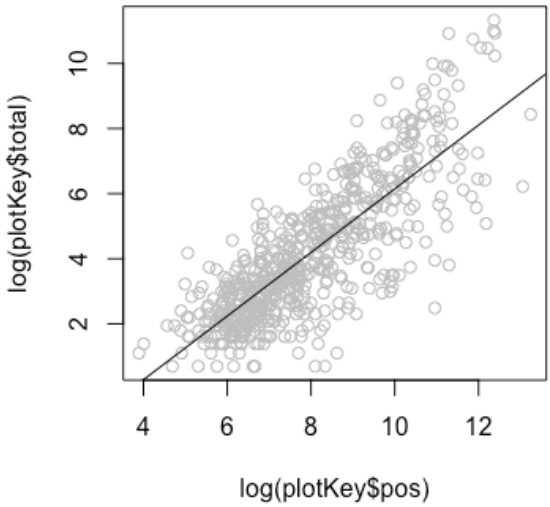
4. An example: comparative adjectives, a huge data set from COCA
    a. Vary between *–er* and *more*
    We might ask: how does frequency affect the choice?
    b. First, is there a relationship between an adjective's non-comparative frequency (pos-freq) and its comparative frequency (comp-freq)?

Ignore the axes below, which are confusing nonsense:

i. x axis = **Pos**itive frequency
ii. y axis = **Comp**arative frequency



5. Fit a line that matches data as closely as possible:
    a. Usually: minimize a cost function
    b. Namely, the distance between the line and each data point, measured as the sum of squared errors (=Ordinary Least Squares method)



6. This line (intercept = -3.63, slope = 0.98) makes a forecast about unseen data, as shown by the equation:

   y = 0.98x –3.63
   Comp = 0.98 * **Pos** – 3.63
   Pos is about 3.6 greater than **Comp**

   An adjective with Pos frequency of 6.21 (that's log frequency, converted to count it's 500) is predicted to have Comp frequency of 2.45 (0.98 * 6.21 – 3.63) (converted to count = 11.5)

7. The example above was **simple linear regression**, we can also do **multiple linear regression**, when we have more than one explanatory variable

8. Example, what's the relationship between **Pos** and **Comp** and **pr(-er)**
   a. Namely: Comp ~ Pos + pr(-er)
   b. In the formula, each explanatory variable gets multiplied by a weight for that variable and added together.
   c. y = w1 * x1 + w2 * x2 … + Intercept
   d. Comp = w1 * Pos + w2 * pr(-er) + Intercept
   e. Values for intercept and ws are called 'co-efficients', essentially the same as weights in Harmonic Grammar, **<u>but</u>**:
      i. explanatory variables can be ***anything*** that's representable with a number (e.g. IQ, hair length, number of toes)
      ii. co-efficients can be negative and/or positive
         1. negative — decreases the dependent variable
         2. positive — increases the dependent variable

9. Model for **Comp:** co-efficients: (note – adding pr(er) didn't change the value of Pos much!)

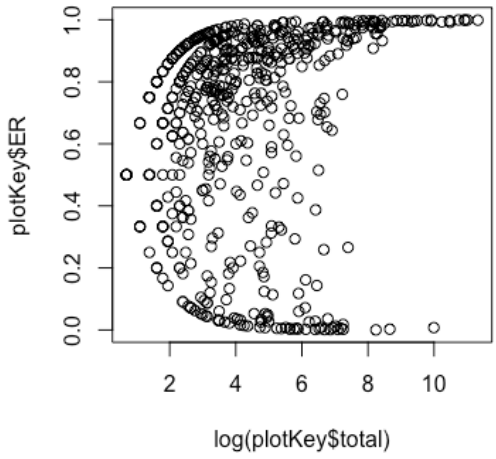
(Intercept)	pr(er)	Pos
-4.4740	1.2456	0.9795

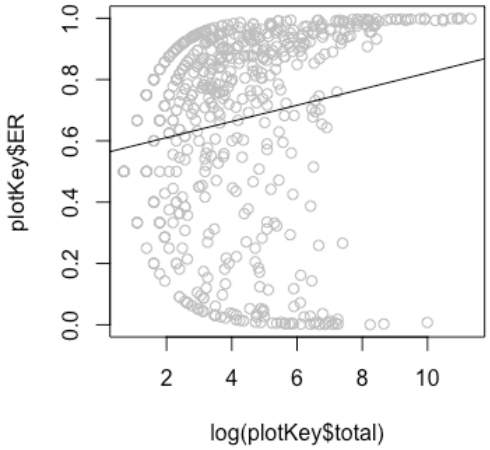
   Predictions about frequency:
   Comp = 0.9795 * **Pos** + 1.2456 * **pr(-er)** – 4.4740

Simple linear regression doesn't work when the dependent variable is a probability

10. Fit a regression for the pr(er) ~ Comp

Again, ignore nonsense axes
y = pr(-er)
x = Comp





Coefficients	(Intercept)	Comp
	0.5576	0.0264

pr(-er) = 0.0264 * Comp + 0.5576

An adjective occurs in the comparative 500 times, how many times will it be –er?

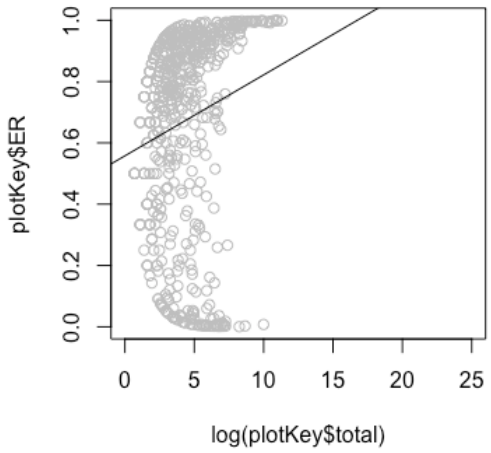
Convert to log frequency… log(500) = 6.2146

pr(-er) = 0.0264 * Comp + 0.5576
pr(-er) = 0.0264 * 6.2146 + 0.5576
pr(-er) = 0.7216654

For an adjective that occurs in the comparative 500 times, we expect 72% of them (361) to be -er

11. **But there's a problem:** Our model predicts that the pr(-er) can be greater than 1!

    When Comp is 16.8 or greater, pr(-er) moves off the graph…



    (Axis labels… y = pr(-er), x = Comp)

12. Solution: **don't fit raw probabilities, fit odds**
    a. Odds of success = number of successes / number of failures
    b. Odds of failure = number of failures / number of successes
    c. prob = odds/(1+odds) 0.50 = 1/(1+1) 0.75 = 3/(3+1)
    d. odds = prob/(1-prob) 1 = 0.50/0.50 3 = .75/.25

13. Horse racing example: in the picture, odds of losing. I've convered them below into odds of winning

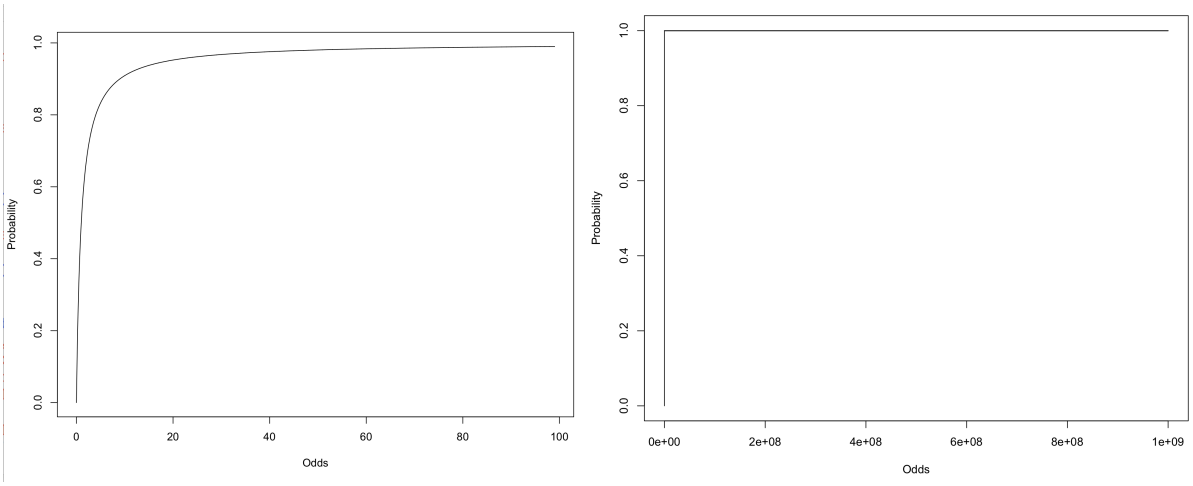
    1:3 = 0.25 1:15 = 0.06 1:5 = 0.17 1:4 = 0.20 1:5 = 0.17
    1:20 = 0.05 1:10 = 0.09 1:5 = 0.17 1:20 = 0.05 1:10 = 0.09
    1:20 = 0.05

14. Trivia: these are directly transformed into payouts (from Horse Racing for Dummies):

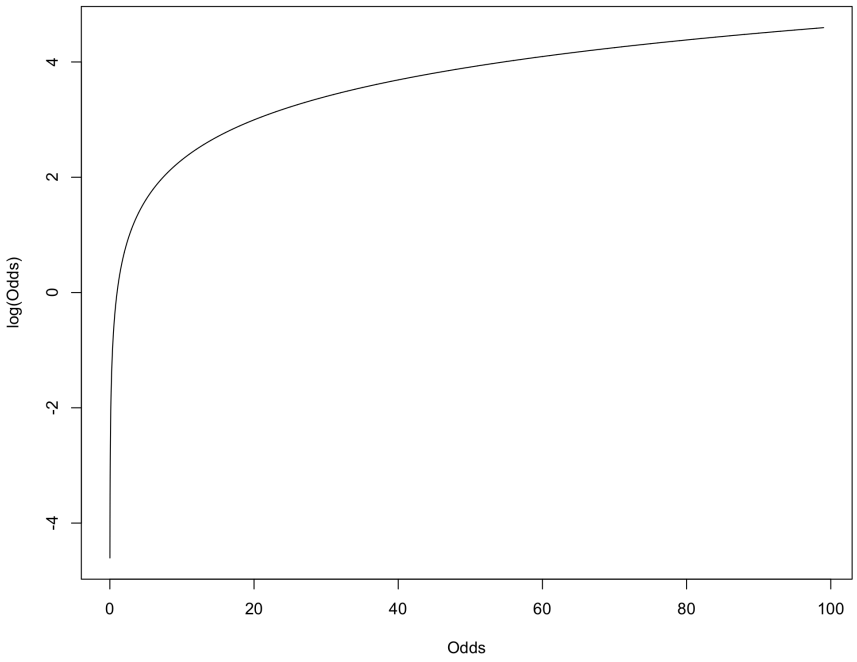
You're betting on horse races and want to know how much your winning bet will give you. To compute your $2 win price, take the odds of your horse and multiply the first number by 2, divide that by the second number, and then add $2 — simple as that! Following is a list of payoffs at various odds for quick reference:

Odds	**$2 Payoff**	**Odds**	**$2 Payoff**	**Odds**	**$2 Payoff**
1/9	$2.20	8/5	$5.20	7/1	$16.00
1/5	$2.40	9/5	$5.60	8/1	$18.00
2/5	$2.80	2/1	$6.00	9/1	$20.00
1/2	$3.00	5/2	$7.00	10/1	$22.00
3/5	$3.20	3/1	$8.00	11/1	$24.00
4/5	$3.60	7/2	$9.00	12/1	$26.00
1/1	$4.00	4/1	$10.00	13/1	$28.00
6/5	$4.40	9/2	$11.00	14/1	$30.00
7/5	$4.80	5/1	$12.00	15/1	$32.00
3/2	$5.00	6/1	$14.00	16/1	$34.00

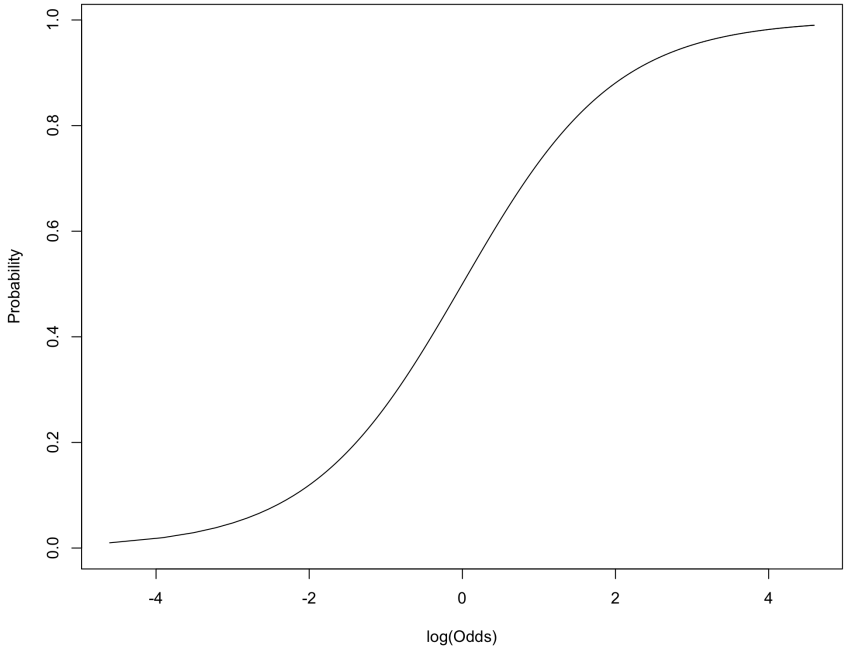
15. A nice property: no matter how much you increase odds, the probability will never exceed 1
    a. Odds of success 1 to 1 = 50%
    b. Odds of success 9 to 1 = 90%
    c. Odds of success 1,000,000 to 1 = 99+%



16. We can make life simpler by taking log(odds)
    a. Simplifies updating models
    b. Gives a nice symmetrical S-shape (sigmoid) where log(odds) of 0 = 50%



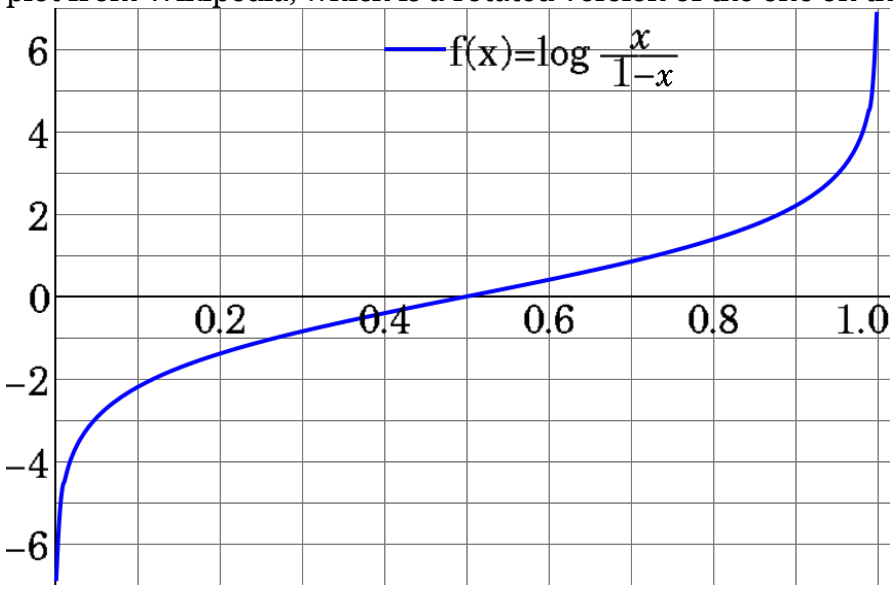
Above: log(Odds) and Odds



Above: log(Odds) and probability

17. Converting from log-odds into probabilities
    a. log-odds to odds : odds = $e^{\text{log-odds}}$
    b. odds to probability: prob = odds/(1+odds)
    c. log-odds to probability: prob = $e^{\text{log-odds}}/(1+ e^{\text{log-odds}})$

18. The function to turn a probability into log-odds is the **logit** function. Here's the plot from Wikipedia, which is a rotated version of the one on the last page



19. A **logistic regression** model is a simply a linear regression model in which the dependent variable is log odds
    a. Also called a **logit model**
    b. This is a GLM = Generalized Linear Model
       A generalization of ordinary linear regression for cases when the response variables aren't normally distributed
    c. Find a set of weights that minimize error between model predictions and observed data

20. The same equation as before, but now all of the coefficients (w1, w2, Intercept) are log-odds

    y = w1 * x1 + w2 * x2 … + Intercept

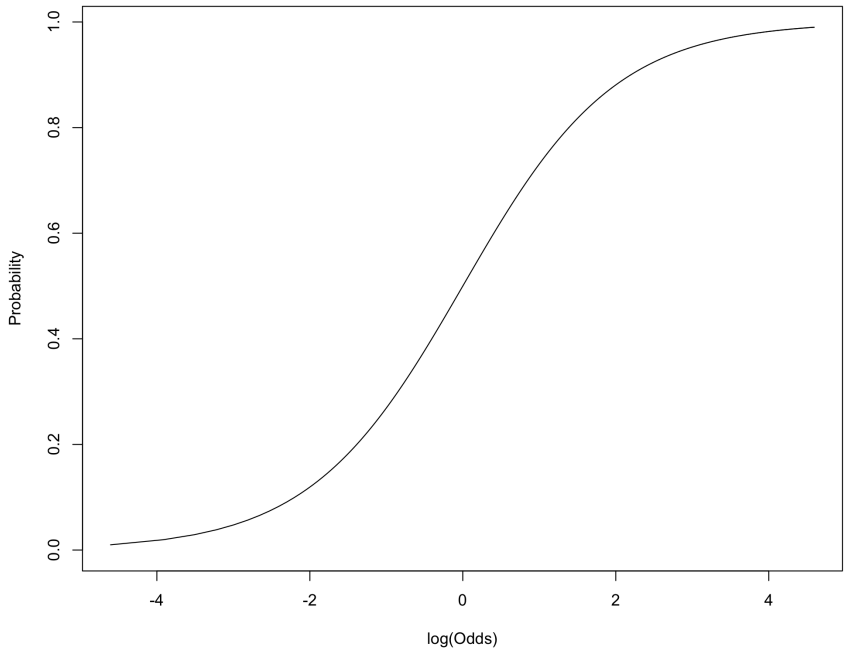
21. First, let's look at a logit model of the *–er* example: pr(–er) ~ Comp

    Coefficients: (Intercept) Comp
    0.5576 0.0264

    log-odds(*-er*) = 0.026 * Comp + 0.5576

    Increasing Comp by 1 increases log-odds(*-er*) by 0.026

    After examining the graph below, it should be clear that an increase of log-odds by 0.026 doesn't cause a constant increase in probability



22. Let's look at the French results from a little while ago, in this case the dependent variable is the log-odds of schwa

	Coefficient
(Intercept)	0.94
Stress = _σ	0.31
Seg = CC_	1.75
Ep/Del = deletion	1.48

I fit this regression by making "violations" –1 or +1:
+1 favors schwa , –1 disfavors schwa

	Stress = _σ	Seg = CC_	Ep/Del = deletion
vest_bleu	+1	+1	–1

This is the same as a comparative vector for the candidates…

	Stress = _σ	Seg = CC_	Faithfulness
vestəbleu	0	0	–1
vest_bleu	–1	–1	0

23. Log-odds of schwa in **veste bleu**

intercept	+ Stress	+ Seg	+ Ep/Del		
0.94	+ 0.31 * 1	+ 1.75 * 1	+ 1.48 * –1	=	1.52

Converting log-odds to probability

prob of schwa = $e^{\text{log-odds}}/(1+ e^{\text{log-odds}})$ prob of no schwa = $1 / (1+ e^{\text{log-odds}})$

0.82 = $e^{1.52}/(1+e^{1.52})$ **0.18** = $1 / (1+e^{1.52})$

24. Visualization

Consider the log-odds and probability graph from earlier. Each of the predictors in the model increases or decreases log-odds by a fixed amount.

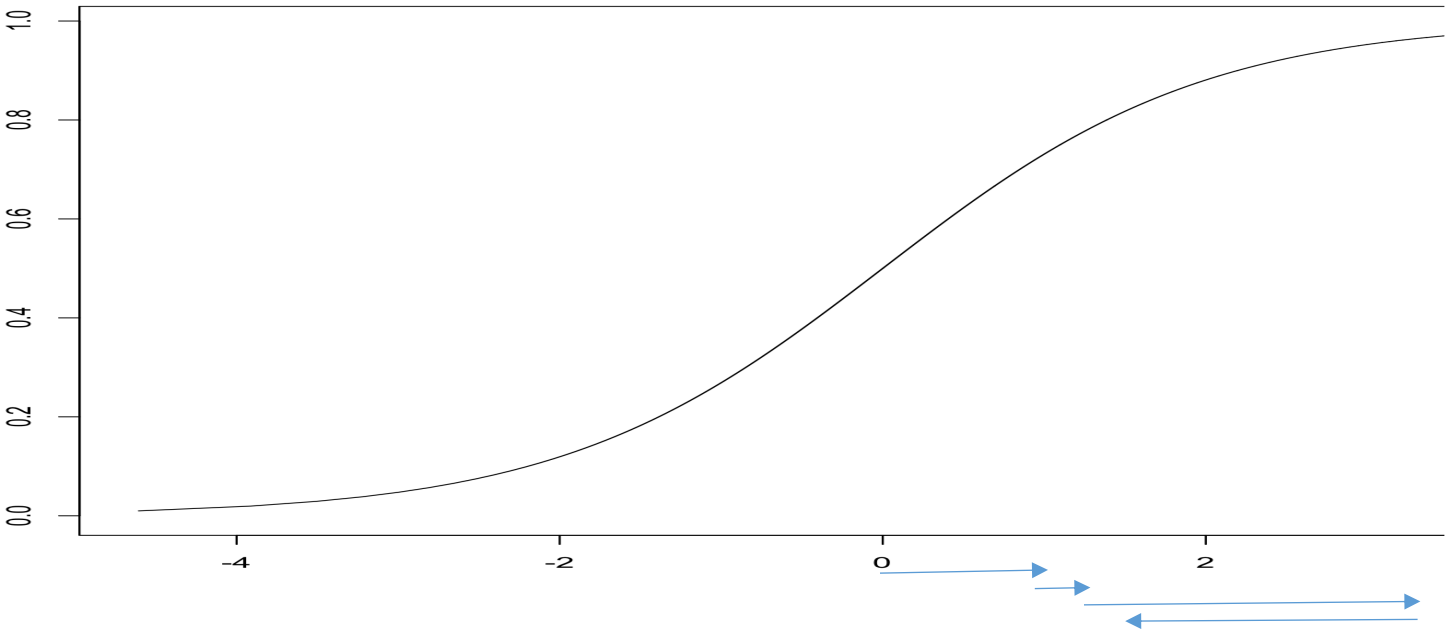
Intercept →

Stress →

Seg →

Ep/Del →

The effect of each predictor on *probability* will depend on where these arrows fall in the graph below. Moving from log-odds +1 to +2 results in a bigger probability increase than moving from log-odds +2 to +3.



25. Probability of schwa in **anie t<u>e</u> disait**

intercept	+ Stress	+ Seg	+ Ep/Del		
0.94	+ 0.31 * –1	+ 1.75 * –1	+ 1.48 * 1	=	0.36

prob = $e^{\text{log-odds}}/(1+ e^{\text{log-odds}})$ prob of no schwa = $1 / (1+ e^{\text{log-odds}})$

0.59 = $e^{0.36}/(1+e^{0.36})$ **0.31** = $1/(1+e^{0.36})$

26. **Q:** So far, how does this look like MaxEnt?

27. **Q:** How is it different?

28. Similarities
    a. For each candidate, we calculate log-odds by summing weighted violations (=Harmony)
    b. For each log-odds score, we convert it into probabilities by applying the exponential function

29. **The revelation:** when we have just two candidates, **the difference between Harmony scores is the log-odds of winning**

	HaveSchwa w= 0.94 (=intercept)	Stress = _σ w=0.31	Seg = CC_ w=1.75	Faithfulness w=1.48 (=Ep/Del)	Harmony
vestəbleu	0	0	0	–1	-1.48
vest_bleu	–1	–1	–1	0	-3
			Candidate A – Candidate B = log-odds of Candidate A		1.52 (cf. 23!)

30. How does this work? Harmony scores 'cancel out'

	HaveSchwa w= 0.94	Stress = _σ w=0.31	Seg = CC_ w=1.75	Faithfulness w=1.48	Harmony
vestəbleu	0	0	0	–1	0
vest_bleu	–1	–1	–1	0	-1.52

31. C1 = harmony of candidate 1
    C2 = harmony of candidate 2

pr(candidate2) = e^{C2} / (e^{C1}+e^{C2}) < Calculating probability in MaxEnt
If C1=0, then e^{C1} = 1 < If harmony of candidate 1 = 0
pr(candidate2) = e^{C2}/ (e^{C2}+1) < The equation to convert from log odds to probability!

32. What this means, practically (given two candidates):
    a. You can look at the difference between candidates to calculate probability
    b. Example: if the two candidates differ in harmony by 2, the more harmonic will have a probability of 88% ($=e^2/(e^2+1)$)

33. How are MaxEnt and Logistic Regression different? Regression models…
    a. Have independent variables, not constraints
        i. Can have negative or positive constraint weights
        ii. Can have any pattern of violations
        iii. Can really be anything (e.g. frequency)
        iv. No theory of constraints for regression
    b. Always model odds of success vs. failure
        v. Need to always choose between two candidates (e.g. schwa vs. no schwa, two allomorphs, etc.)
        vi. Multinomial logistic regression, not covered, is a model that allows more candidates, compares every candidate to a baseline and normalizes
        vii. You need the candidates to be parallel across candidate sets ('success' is always defined in the same way)
    c. Have intercepts
        viii. 'baseline' preference between two outcomes
        ix. MaxEnt has something similar, by weighting constraints that prefer each outcome
            1. For schwa, *Schwa and Max(schwa)
            2. For –er and *more*, we could imagine constraints like 'Use –er' and 'Use more'