Classification

CMSC 473/673 - NATURAL LANGUAGE PROCESSING

Slides modified from Dr. Frank Ferraro

Learning Objectives

Model classification problems using logistic regression

Define appropriate features for a logistic regression problem

Define an objective for LR modeling

Visualize the learning process for maxent models

Distinguish between discriminatively- and generatively-trained maxent models

Outline

Maximum Entropy classifiers

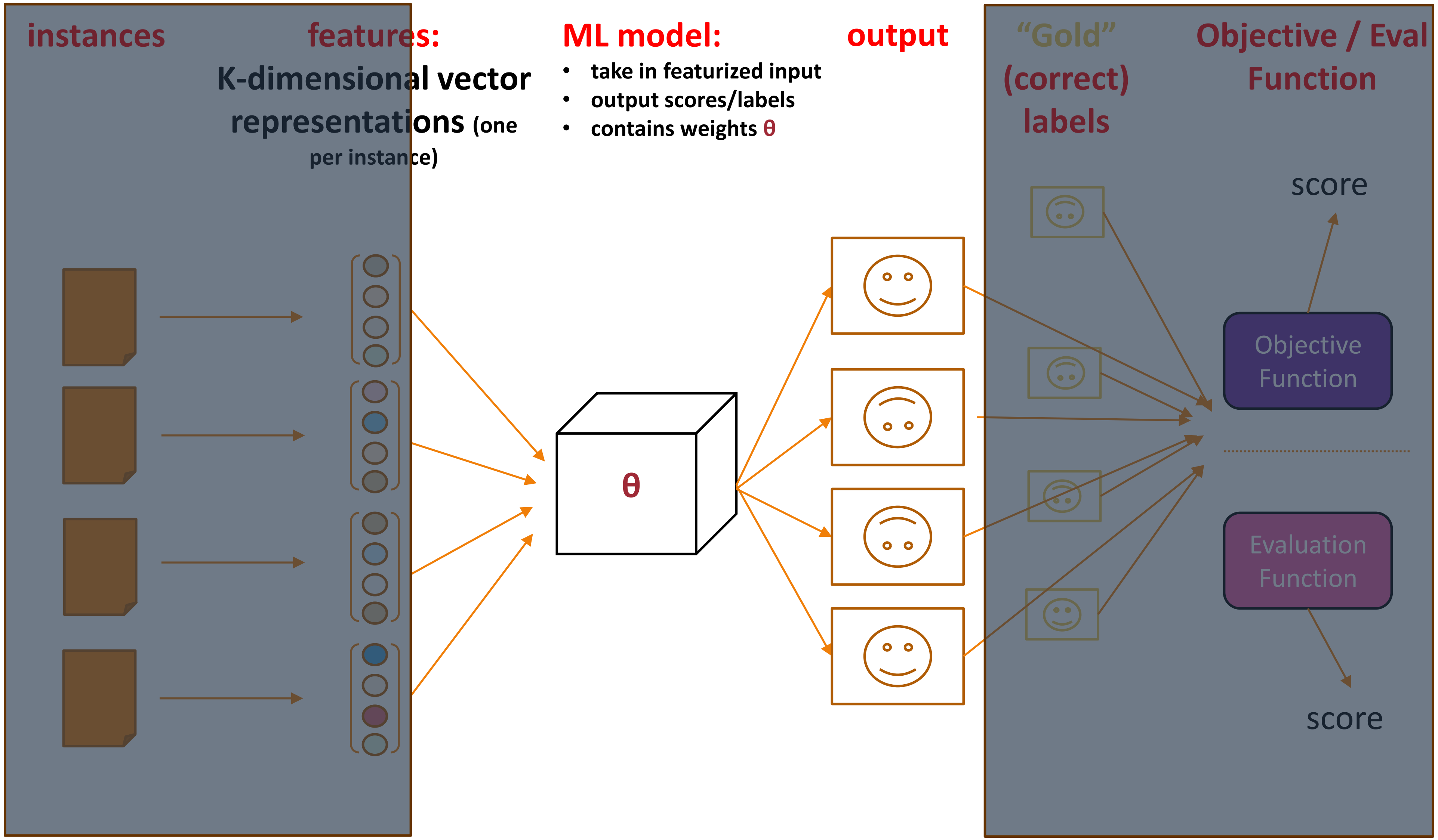
- Defining the model: Discriminatively
- Defining the objective
- Learning: Optimizing the objective
- Defining the model: Generatively

Outline

Maximum Entropy classifiers

- Defining the model: Discriminatively
- Defining the objective
- Learning: Optimizing the objective
- Defining the model: Generatively

Defining the Model



Examining Assumption 3 Made for Classification Evaluation

Given X, our classifier produces a score for each possible label

p(●|X) vs. p(○|X)

$$\text{best label} = \underset{\text{label}}{\arg\max}\ P(\text{label}|\text{example})$$

💡 Key Take-away 💡

We will ***learn*** this

$$\boldsymbol{p(Y \mid X)}$$

Conditional probability:
probability of event Y, assuming event X happens too

NLP pg. 477

Maxent Models for Classification: Discriminatively or …

Directly model the posterior

$$p(Y \mid X) = \mathbf{maxent}(X; Y)$$

Discriminatively trained classifier

"Discriminative classifiers like logistic regression instead learn what features from the input are most useful to discriminate between the different possible classes."
SLP, ch. 4

Bayes' Rule

$$\underbrace{P(Y|X)}_{\text{Posterior}} = \frac{\overbrace{P(X|Y)}^{\text{Likelihood}} \cdot \overbrace{P(Y)}^{\text{Prior}}}{P(X)}$$

Posterior:
probability of event Y with knowledge that X has occurred
NLP pg. 478

Likelihood:
probability of event X given that Y has occurred
NLP pg. 478

Prior:
probability of event X occurring (regardless of what other events happen)
NLP pg. 478

Terminology: Posterior Probability

Posterior probability:

$$p(\,Y = label_1\,|X) \text{ vs. } p(Y = label_0\,|X)$$

Conditionally dependent probabilities:

- If $label_0$ and $label_1$ are the only two options:

$$p(\,Y = label_1\,|X) + p(Y = label_0\,|X) = 1$$

and

$$p(\,Y = label_1\,|X) \geq 0,\; p(Y = label_0\,|X) \geq 0$$

Maxent Models for Classification: Discriminatively or Generatively Trained

Directly model the posterior

$$p(Y \mid X) = \mathbf{maxent}(X; Y)$$

Discriminatively trained classifier

Model the posterior with Bayes rule

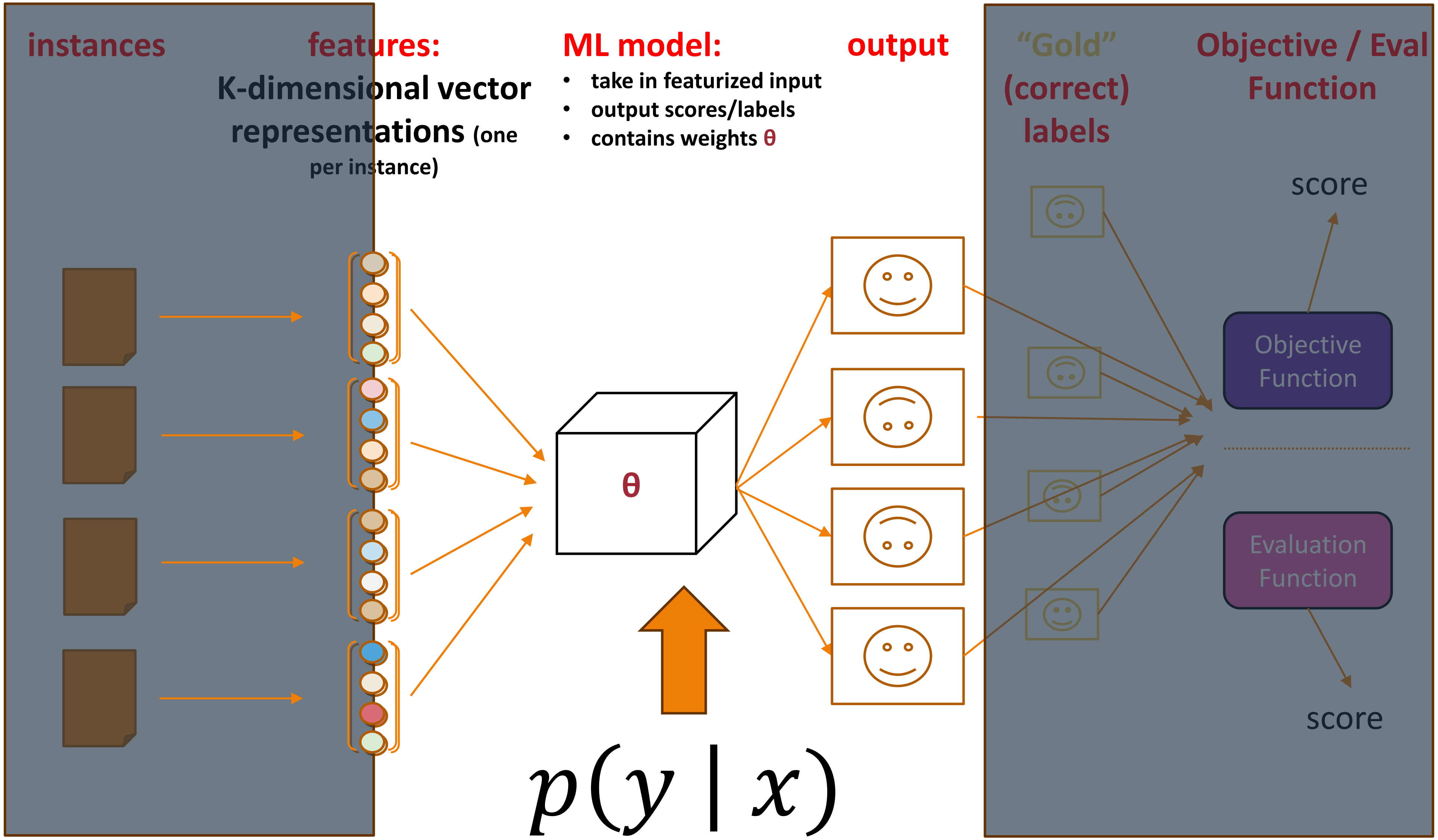
$$p(Y \mid X) \propto \mathbf{maxent}(X \mid Y)p(Y)$$

Generatively trained classifier with maxent-based language model

Maximum Entropy (Log-linear) Models For Discriminatively Trained Classifiers

$$p(y \mid x) = \text{maxent}(x, y)$$

Modeled jointly!



$$p(y \mid x) = \text{maxent}(x, y)$$

Core Aspects to Maxent Classifier p(y|x)

We need to define:

- **features** $f(x)$ from x that are meaningful;
- **weights** θ (at least one per feature, often one per feature/label combination) to say how important each feature is; and
- a way to **form probabilities** from f and θ

Overview of Featurization

Common goal: probabilistic classifier p(y | x)

Often done by defining **features** between x and y that are meaningful

- Denoted by a **general vector of K features**

$$f(x) = (f_1(x), \dots, f_K(x))$$

Features can be thought of as "soft" rules

- E.g., POSITIVE sentiments tweets *may* be more likely to have the word "happy"

Discriminative Document Classification

s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.

h: The Bulls basketball team is based in Chicago.

What does it mean for a feature to “fire”?

We need to *score* the different extracted clues.

$\text{extract\_and\_score}_{\text{Bulls, entailed}}(\text{🗎})$ **ENTAILED**

Jackson and the star cast,
including Scottie Pippen, took
the Chicago Bulls to six
National Basketball
Association championships.

$\text{extract\_and\_score}_{\text{basketball, entailed}}(\text{🗎}, \text{ENTAILED})$

h: The Bulls basketball team
is based in Chicago.

$\text{extract\_and\_score}_{\text{Chicago, entailed}}(\text{🗎}, \text{ENTAILED})$

Score and Combine Our Clues

$\text{score}_{1,\text{ Entailed}}(\text{🗎})$

$\text{score}_{2\text{ , Entailed}}(\text{🗎})$

$\text{score}_{3\text{ , Entailed}}(\text{🗎})$

…

$\text{score}_{k\text{ , Entailed}}(\text{🗎})$

…

COMBINE

posterior probability of ENTAILED

Scoring Our Clues

$$\text{score}\Big(\ \boxed{\begin{array}{l}\text{s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the}\\ \text{Chicago Bulls to six National Basketball Association championships.}\\ \text{h: The Bulls basketball team is based in Chicago.}\end{array}}\ , \text{ENTAILED}\Big) =$$

(ignore the feature indexing for now)

$\text{score}_{1\text{, Entailed}}(🗎)$ ✚

$\text{score}_{2\text{, Entailed}}(🗎)$ ✚

$\text{score}_{3\text{, Entailed}}(🗎)$ ✚

…

Turning Scores into Probabilities

score(s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago. , ENTAILED) > score(s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago. , NOT ENTAILED)

p(ENTAILED | s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.) > p(NOT ENTAILED | s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.)

KEY IDEA

Turning Scores into Probabilities (More Generally)

$$\text{score}(x, y_1) > \text{score}(x, y_2)$$

$$\Downarrow$$

$$p(y_1 | x) > p(y_2 | x)$$

KEY IDEA

Maxent Modeling

$$\underbrace{p(\text{ENTAILED} \mid \text{s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships. h: The Bulls basketball team is based in Chicago.})}_{\text{This must be a probability}} \propto$$

This could be any real number

Convert through function G? What is this function?

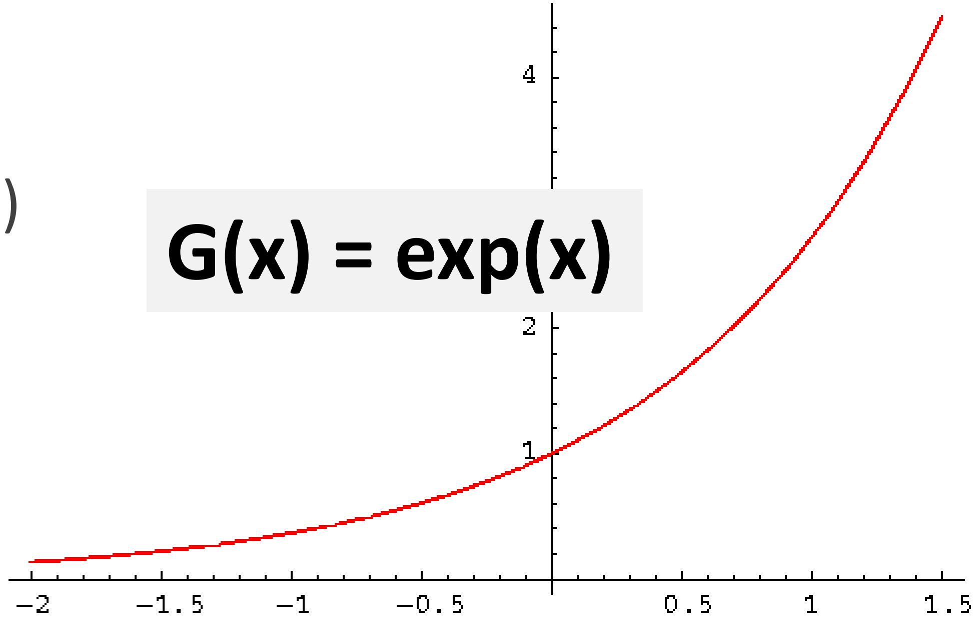
$$\overbrace{G}\overbrace{(\text{score}(\text{s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships. h: The Bulls basketball team is based in Chicago.}, \text{ENTAILED}))}$$

What function G…

operates on any real number?

is never less than 0?

is monotonic? (a < b ➔ G(a) < G(b))



Maxent Modeling

p(ENTAILED | s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.) ∝

exp($\text{weight}_{1,\text{ Entailed}} * \text{applies}_1(🗎)$ + $\text{weight}_{2,\text{ Entailed}} * \text{applies}_2(🗎)$ + $\text{weight}_{3,\text{ Entailed}} * \text{applies}_3(🗎)$ + …))

K different weights… for K different features

$$\theta = \begin{bmatrix} .31 \\ -.5 \\ .1 \\ .002 \\ .522 \\ \dots \end{bmatrix} \qquad f(x) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ \dots \end{bmatrix}$$

Maxent Modeling

$$p\Big(\text{ENTAILED} \;\Big|\; \boxed{\begin{array}{l}\text{s: Michael Jordan, coach Phil Jackson and the star cast,}\\ \text{including Scottie Pippen, took the Chicago Bulls to six}\\ \text{National Basketball Association championships.}\\ \text{h: The Bulls basketball team is based in Chicago.}\end{array}}\Big) \propto$$

$$\exp\Big(\begin{array}{l} \text{weight}_{1,\ \text{Entailed}} * \text{applies}_1(🗎)\ + \\ \text{weight}_{2,\ \text{Entailed}} * \text{applies}_2(🗎)\ + \\ \text{weight}_{3,\ \text{Entailed}} * \text{applies}_3(🗎)\ + \\ \quad \ldots \end{array}\Big)\Big)$$

K different weights…

for K different features

multiplied and then summed

Maxent Modeling

p(ENTAILED | s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.) ∝

exp(Dot_product of Entailed weight_vec feature_vec(🗎))

K different weights…

for K different features

multiplied and then summed

Maxent Modeling

$$p(\textsc{Entailed} \mid \text{s, h}) \propto$$

s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.

$$\begin{bmatrix} .31 & -.5 & .1 & .002 & .522 & \dots \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ \dots \end{bmatrix}$$

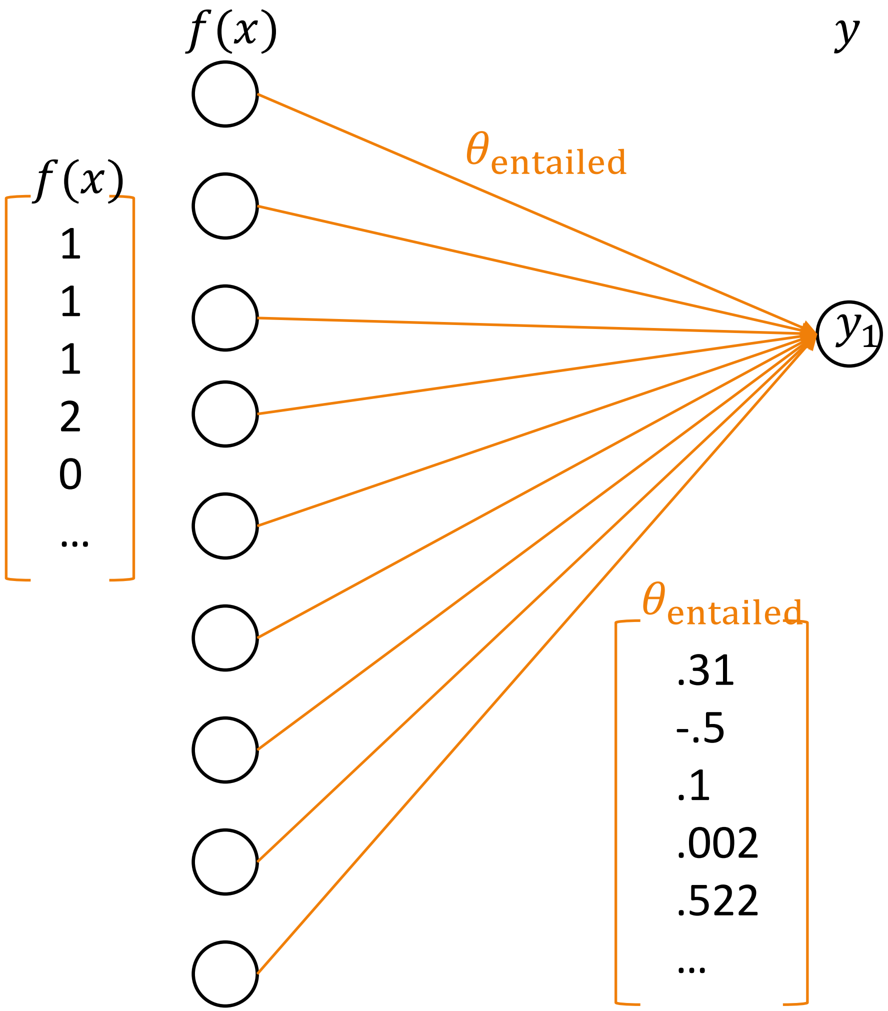
$$\exp(\theta^{T}_{\text{ENTAILED}} f(\ \ ))$$

K different weights…

for K different features

multiplied and then summed

Maxent Classifier, schematically



$p(y = \text{entailed}|x) \propto \exp(\theta_{\text{entailed}} f(x))$

Maxent Modeling

$p(\text{ENTAILED} \mid$

s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.

$) \propto$

$\frac{1}{Z}\exp(\theta_{\text{ENTAILED}}^{T} f($  $))$

Maxent Modeling

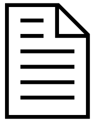
$$p(\text{ENTAILED} \mid \text{s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships. h: The Bulls basketball team is based in Chicago.}) =$$

s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.

$$\frac{1}{Z}\exp(\theta_{\text{ENTAILED}}^{T} f(\cdot))$$

How do we define Z?

Normalization for Classification

$Z = \sum_{\text{label } j} \exp(\theta_j^T f($  $))$

$p(y \mid x) \propto \exp(\theta_y^T f(x))$ *classify doc x with label y in one go*

Normalization for Classification (long form)

$$Z =$$

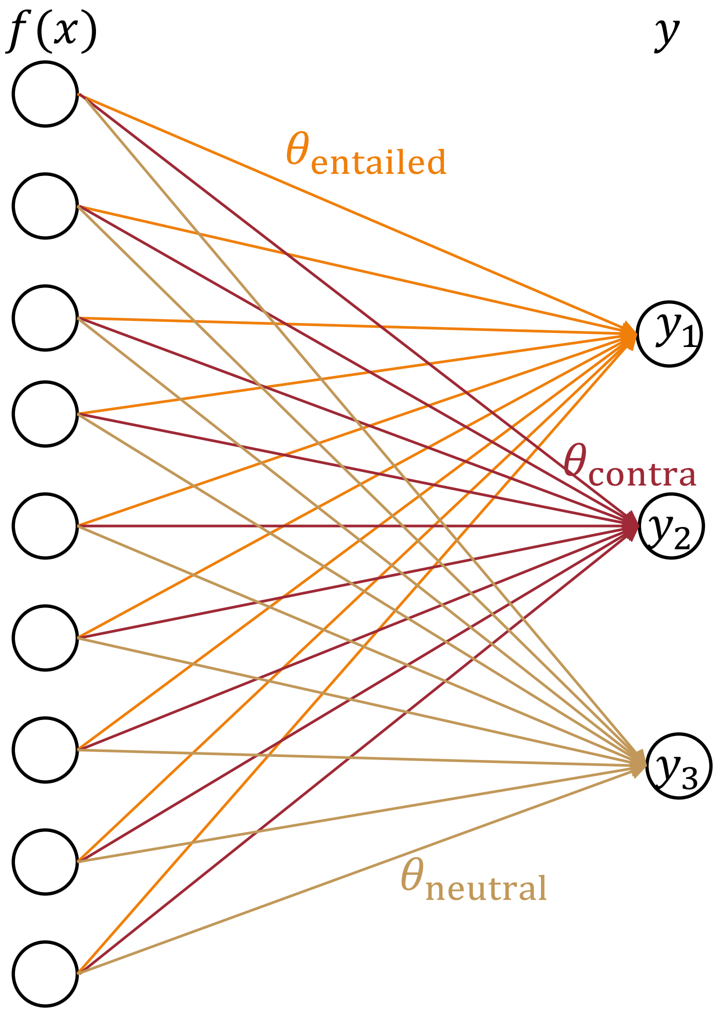
$$\sum_{\text{label } j} \exp\left(\begin{array}{l} \text{weight}_{1,j} * \text{applies}_1(\,) + \\ \text{weight}_{2,j} * \text{applies}_2(\,) + \\ \text{weight}_{3,j} * \text{applies}_3(\,) + \\ \quad\quad \dots \end{array} \right)$$

$$p(y \mid x) \propto \exp(\theta_y^T f(x))$$

classify doc x with label y in one go

Maxent Classifier, schematically

Why would we want to normalize the weights?



$p(y = \text{entailed}|x) \propto \exp(\theta_{\text{entailed}} f(x))$

$p(y = \text{contra}|x) \propto \exp(\theta_{\text{contra}} f(x))$

$p(y = \text{neutral}|x) \propto \exp(\theta_{\text{neutral}} f(x))$

output:
i = argmax score$_i$
class i

Final Equation for Logistic Regression

features $f(x)$ from x that are meaningful;

weights θ (at least one per feature, often one per feature/label combination) to say how important each feature is; and

a way to **form probabilities** from f and θ

$$p(y \mid x) = \frac{\exp(\theta_y^T f(x))}{\sum_{y'} \exp(\theta_{y'}^T f(x))}$$

Different Notation, Same Meaning

$$p(Y = y \mid x) = \frac{\exp(\theta_y^T f(x))}{\sum_{y'} \exp(\theta_{y'}^T f(x))}$$

$$p(Y = y \mid x) \propto \exp(\theta_y^T f(x))$$

$$p(Y \mid x) = \mathrm{softmax}(\theta f(x))$$

Defining Appropriate Features in a Maxent Model

Feature functions help extract useful features (characteristics) of the data

They turn *data* into *numbers*

Features that are not 0 are said to have fired

Generally *templated*

Binary-valued (0 or 1) or real-valued

Representing Linguistic Information

User-defined	Integer representation/one-hot encoding	Assign each word to some index i, where $0 \leq i < V$ Represent each word w with a V-dimensional **binary** vector e_w, where $e_{w,i} = 1$ and 0 otherwise
Model-produced	Dense embedding	Let E be some *embedding size* (often 100, 200, 300, etc.) Represent each word w with an E-dimensional **real-valued** vector e_w

Featurization is Similar but…

Vocab types (V) / embedding dimension (E) ➔ number of features (number of "clues")

"Linguistic blob" ➔ Instances to represent

Features are extracted on each instance

Review: Bag-of-words as a Function

Based on some tokenization, turn an input document into an array (or dictionary or set) of its unique vocab items

Think of getting a BOW rep. as a function f

input: Document

output: Container of size E, indexable by each vocab type v

Some Bag-of-words Functions

Kind	Type of f_v	Interpretation
Binary	0, 1	Did *v* appear in the document?
Count-based	Natural number (int >= 0)	How often did *v* occur in the document?
Averaged	Real number (>=0, <= 1)	How often did *v* occur in the document, normalized by doc length?
TF-IDF (term frequency, inverse document frequency)	Real number (>= 0)	How frequent is a word, tempered by how prevalent it is across the corpus (to be covered later!)
…		

Q: Is this a reasonable representation?

Q: What are some tradeoffs (benefits vs. costs)?

Useful Terminology: n-gram

Within a larger string (e.g., sentence),
a contiguous sequence of n items (e.g., words)

Colorless green ideas sleep furiously

n	Commonly called	History Size (Markov order)	Example n-gram ending in "furiously"
1	unigram	0	furiously
2	bigram	1	sleep furiously
3	trigram (3-gram)	2	ideas sleep furiously
4	4-gram	3	green ideas sleep furiously
n	n-gram	n-1	$w_{i-n+1} \ldots w_{i-1}\ w_i$

Templated Features

Define a feature fclue(🖹) for each clue you want to consider

The feature fclue fires if the clue applies to/can be found in 🖹

Clue is often a target phrase (an n-gram)

Maxent Modeling:
Templated Binary Feature Functions

p(ENTAILED |

s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.
h: The Bulls basketball team is based in Chicago.

) ∝

exp(
weight$_{1\text{, Entailed}}$ * applies$_1$(🗎) +
weight$_{1\text{, Entailed}}$ * applies$_2$(🗎) +
weight$_{1\text{, Entailed}}$ * applies$_3$(🗎) +
…
))

$$\text{applies}_{\text{target}}(🗎) = \begin{cases} 1, \text{target } matches \text{ 🗎} \\ 0, \quad \text{otherwise} \end{cases}$$

binary

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(\text{🖹}) = \begin{cases} 1, \text{target } \mathit{matches} \text{ 🖹} \\ 0, \qquad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(\text{🖹}) = \begin{cases} 1, \text{ball } \mathit{in} \text{ both s and h of 🖹} \\ 0, \qquad \text{otherwise} \end{cases}$$

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(\text{🗎}) = \begin{cases} 1, \text{target } \mathit{matches} \text{ 🗎} \\ 0, \quad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(\text{🗎}) = \begin{cases} 1, \text{ball } \mathit{in} \text{ both s and h of 🗎} \\ 0, \quad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(\text{🗎}) = \begin{cases} 1, \text{target } \mathit{matches} \text{ 🗎} \\ 0, \quad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(\text{🗎}) = \begin{cases} 1, \text{ball } \mathit{in} \text{ both s and h of 🗎} \\ 0, \quad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

A1: VL

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(🗎) = \begin{cases} 1, \text{target } matches \text{ 🗎} \\ 0, \qquad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(🗎) = \begin{cases} 1, \text{ball } in \text{ both s and h of 🗎} \\ 0, \qquad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

A1: VL

2. How many features are defined if bigram targets are used?

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(\text{🖹}) = \begin{cases} 1, \text{target } \mathit{matches}\ \text{🖹} \\ 0, \quad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}\,(\text{🖹}) = \begin{cases} 1, \text{ball } \mathit{in} \text{ both s and h of } \text{🖹} \\ 0, \quad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

A1: VL

2. How many features are defined if bigram targets are used (w/ each label)?

A2: V^2L

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(🗎) = \begin{cases} 1, \text{target } \textit{matches } 🗎 \\ 0, \qquad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(🗎) = \begin{cases} 1, \text{ball } \textit{in } \text{both s and h of } 🗎 \\ 0, \qquad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

A1: VL

2. How many features are defined if bigram targets are used (w/ each label)?

A2: V^2L

3. How many features are defined if unigram and bigram targets are used (w/ each label)?

Example of a Templated Binary Feature Functions

$$\text{applies}_{\text{target}}(\text{🗎}) = \begin{cases} 1, \text{target } \mathit{matches} \text{ 🗎} \\ 0, \quad\quad \text{otherwise} \end{cases}$$

$$\text{applies}_{\text{ball}}(\text{🗎}) = \begin{cases} 1, \text{ball } \mathit{in} \text{ both s and h of 🗎} \\ 0, \quad\quad \text{otherwise} \end{cases}$$

Q: If there are V vocab types and L label types:

1. How many features are defined if unigram targets are used (w/ each label)?

A1: VL

2. How many features are defined if bigram targets are used (w/ each label)?

A2: V^2L

3. How many features are defined if unigram and bigram targets are used (w/ each label)?

A2: $(V + V^2)L$

Outline

Maximum Entropy classifiers

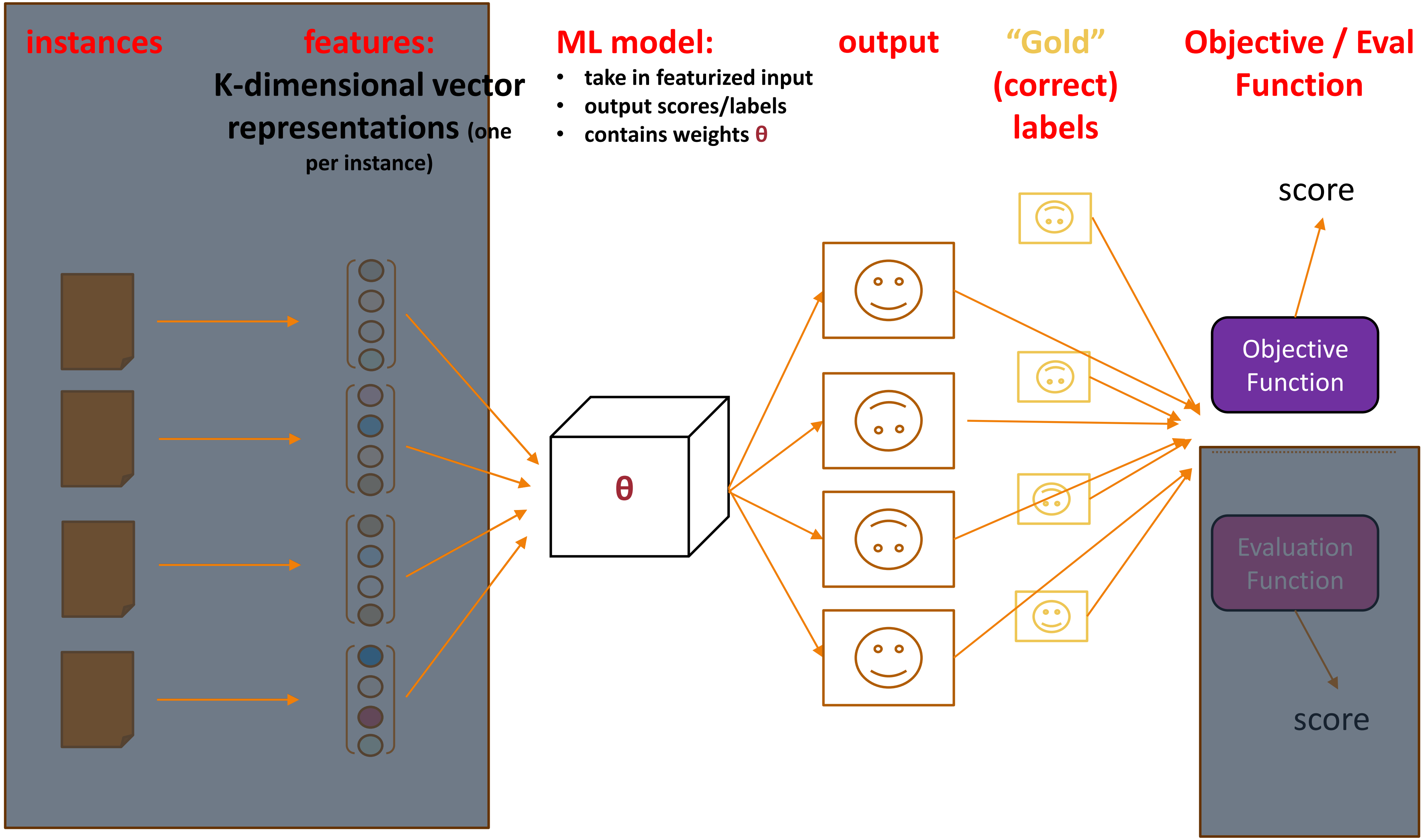
- Defining the model: Discriminatively
- Defining the objective
- Learning: Optimizing the objective
- Defining the model: Generatively

$$p_\theta(y \mid x)$$ probabilistic model

$$\Downarrow$$

$$F(\theta; x, y)$$ **objective**

Defining the Objective



Primary Objective: Likelihood

Goal: *maximize* the score your model gives to the training data it observes

This is called the **likelihood of your data**

In classification, this is p(label | 🗎)

For language modeling, this is p(word | history of words)

Objective = Full Likelihood? (Classification)

$$\prod_i \overbrace{p_\theta(y_i|x_i)}^{\text{Our goal probability}} \propto \prod_i \overbrace{\exp(\theta_{y_i}^T f(x_i))}^{\text{Our maxent equation}}$$

These values can have very small magnitude ➔ underflow

Differentiating this product could be a pain

Logarithms

$(0, 1] \rightarrow (-\infty, 0]$

Products ➔ Sums

log(ab) = log(a) + log(b)

log(a/b) = log(a) – log(b)

Inverse of exp

log(exp(x)) = x

How might you find the log of this?

$$\prod_i p_\theta(y_i|x_i)$$

Log-Likelihood (Classification)

Wide range of (negative) numbers
Sums are more stable

$$\log \prod_i p_\theta(y_i|x_i) = \sum_i \log p_\theta(y_i|x_i)$$

Products ➔ Sums

log(ab) = log(a) + log(b)

log(a/b) = log(a) – log(b)

Maximize Log-Likelihood (Classification)

Original maxent equation

$$\frac{\exp(\theta_y^T f(x))}{\sum_{y'} \exp(\theta_{y'}^T f(x))}$$

$$\log \prod_i p_\theta(y_i|x_i) = \sum_i \log p_\theta(y_i|x_i)$$

Differentiating this becomes nicer (even though Z depends on θ)

Inverse of exp

log(exp(x)) = x

$$= \sum_i \overbrace{\theta_{y_i}^T f(x_i) - \log Z(x_i)}$$

Log-Likelihood (Classification)

Wide range of (negative) numbers
Sums are more stable

$$\log \prod_i p_\theta(y_i | x_i) = \sum_i \log p_\theta(y_i | x_i)$$

$$= \sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i)$$

$$= F(\theta)$$

Equivalent Version 2:
Minimize Cross Entropy Loss

loss uses y (random variable), or model's probabilities $\ell^{\text{xent}}\left(\overrightarrow{y^*}, p(y|x)\right)$

$$\ell^{\text{xent}}\left(\overrightarrow{y^*}, y\right)$$

index of "1" indicates correct value

$$\begin{pmatrix} 0 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{pmatrix}$$

one-hot vector

Cross entropy:
How much $\hat{y}$ differs from the true y

objective is convex
(when f(x) is not learned)

Equivalent Version 2: *Minimize* Cross Entropy Loss

loss uses y (random variable), or model's probabilities $\ell^{\text{xent}}\left(\overrightarrow{y^*}, p(y|x)\right)$

$$\ell^{\text{xent}}\left(\overrightarrow{y^*}, y\right) = -\sum_k \overrightarrow{y^*}[k] \log p(y = k|x)$$

index of "1" indicates correct value

$$\begin{pmatrix} 0 \\ 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{pmatrix}$$

one-hot vector

Classification Log-likelihood ≅ Cross Entropy Loss

$$F(\theta) = \sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i)$$

CROSSENTROPYLOSS

CLASS `torch.nn.CrossEntropyLoss(weight=None, size_average=None, ignore_index=-100, reduce=None, reduction='mean')` [SOURCE]

This criterion combines `LogSoftmax` and `NLLLoss` in one single class.

It is useful when training a classification problem with C classes. If provided, the optional argument `weight` should be a 1D *Tensor* assigning weight to each of the classes. This is particularly useful when you have an unbalanced training set.

The *input* is expected to contain raw, unnormalized scores for each class.

input has to be a Tensor of size either $(minibatch, C)$ or $(minibatch, C, d_1, d_2, ..., d_K)$ with $K \geq 1$ for the *K*-dimensional case (described later).

This criterion expects a class index in the range $[0, C-1]$ as the *target* for each value of a 1D tensor of size *minibatch*; if *ignore_index* is specified, this criterion also accepts this class index (this index may not necessarily be in the class range).

The loss can be described as:

$$\text{loss}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right)$$

Preventing Extreme Values

Likelihood on its own can lead to overfitting and/or extreme values in the probability computation

$$F(\theta) = \sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i)$$

Learn the parameters based on some (fixed) data/examples

Regularization: Preventing Extreme Values

$$F(\theta) = \sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i)$$

With fixed/predefined features, the values of θ determine how “good” or “bad” our objective learning is

Regularization: Preventing Extreme Values

$$F(\theta) = \left(\sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i) \right) - R(\theta)$$

With fixed/predefined features, the values of θ determine how “good” or “bad” our objective learning is

- Augment the objective with a **regularizer**
  - This regularizer places an inductive bias (or, prior) on the general “shape” and values of θ

(Squared) L2 Regularization

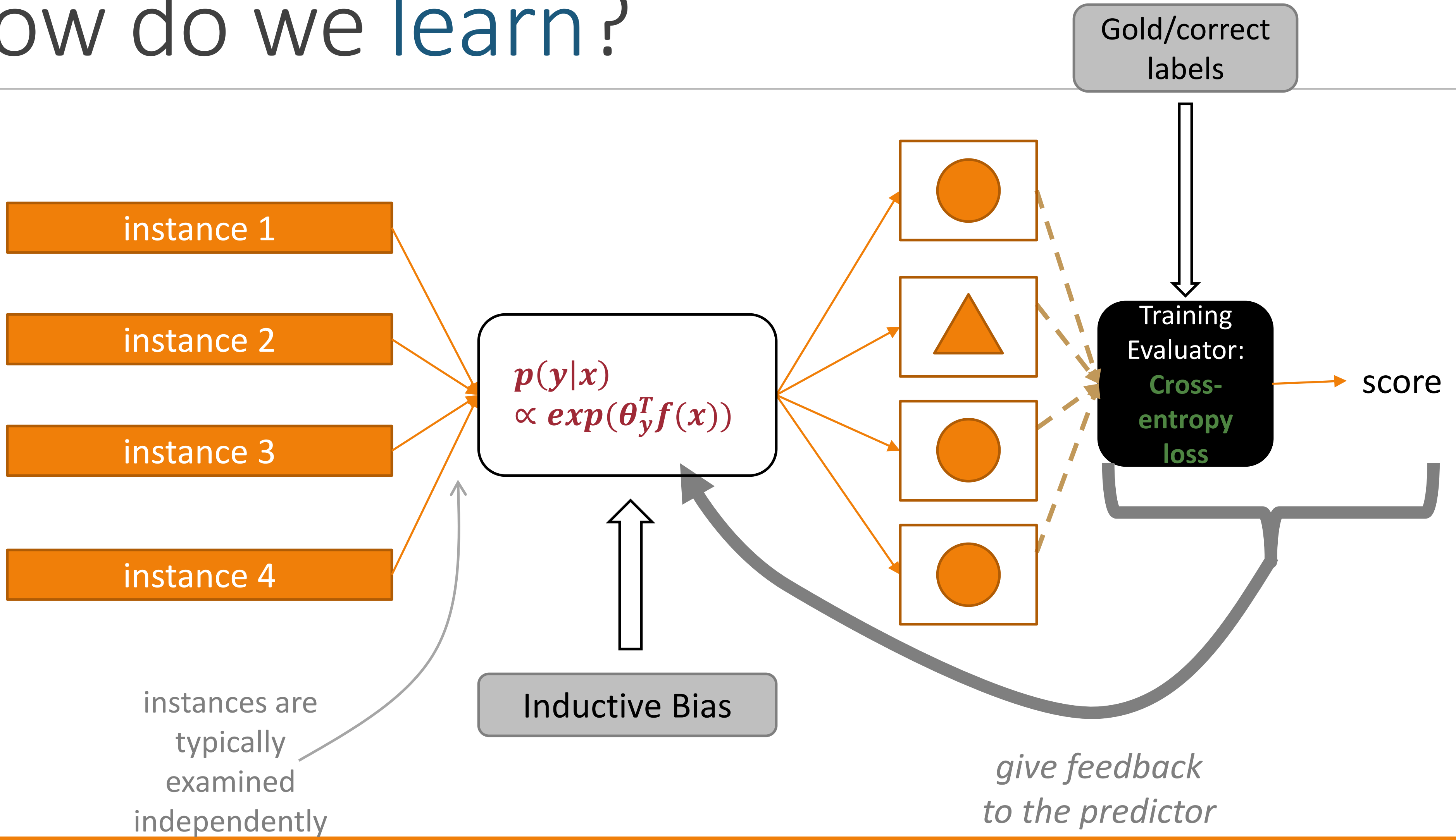
$$R(\theta) = \|\theta\|_2^2 = \sum_k \theta_k^2$$

Outline

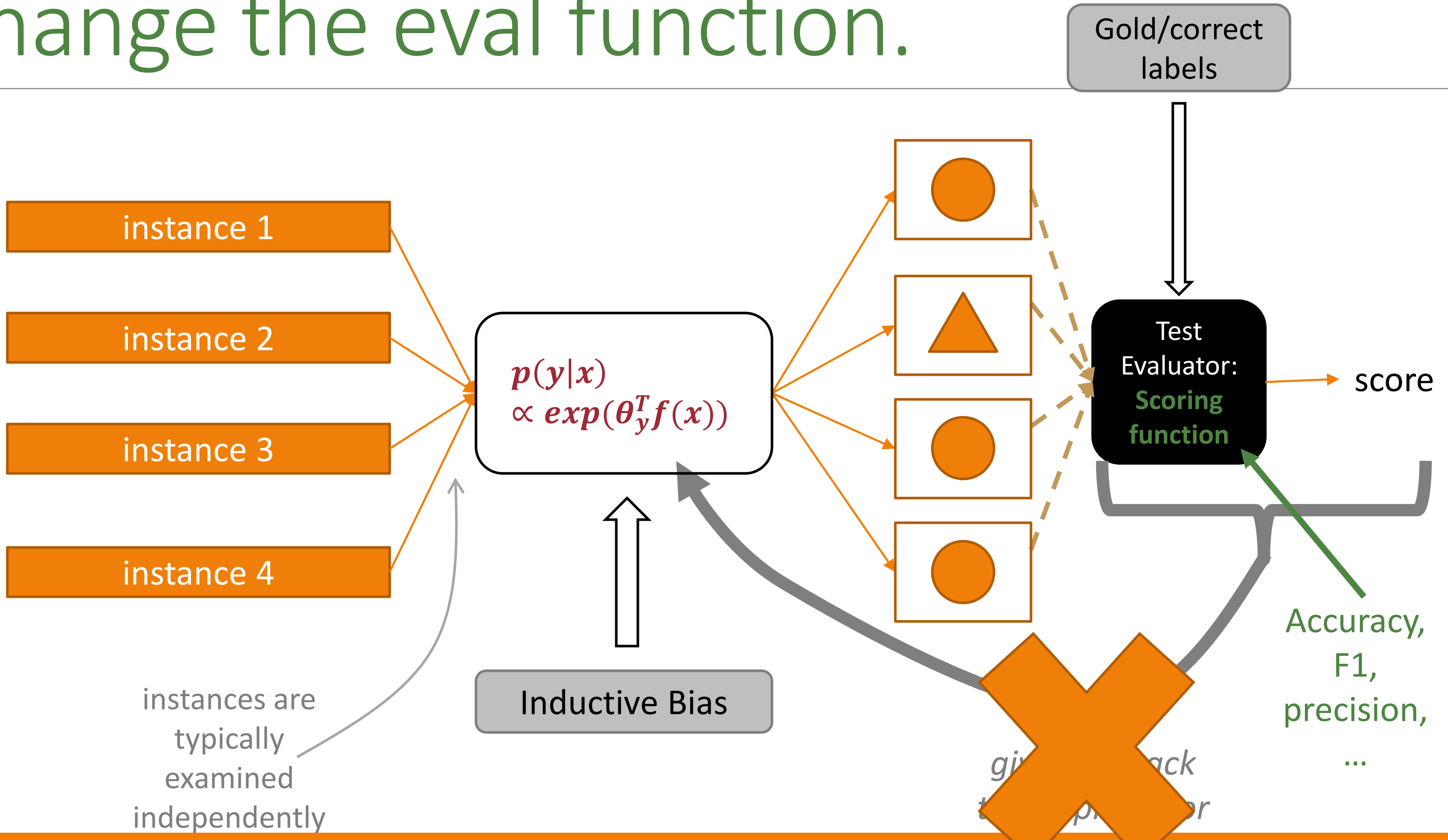
Maximum Entropy classifiers

- Defining the model: Discriminatively
- Defining the objective
- Learning: Optimizing the objective
- Defining the model: Generatively

How do we learn?

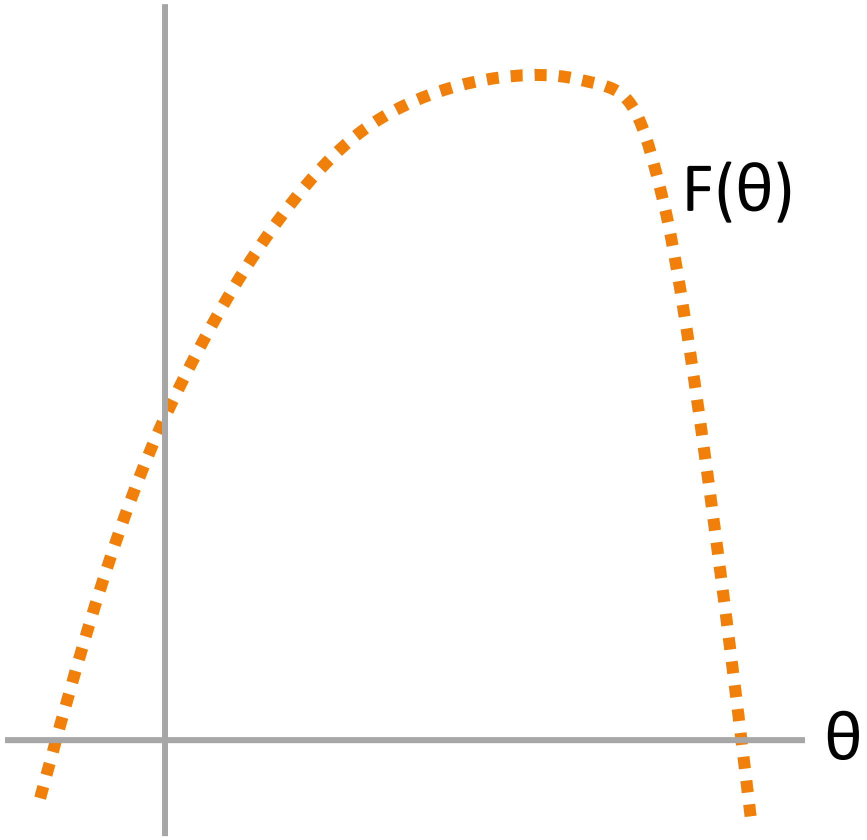


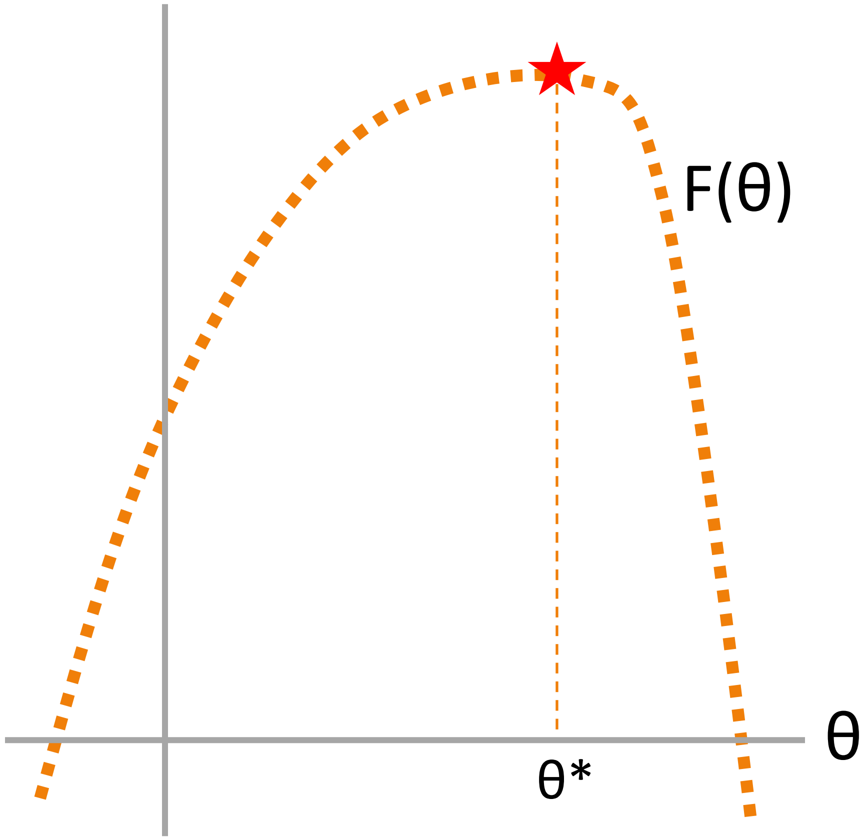
How do we evaluate (or use)? Change the eval function.

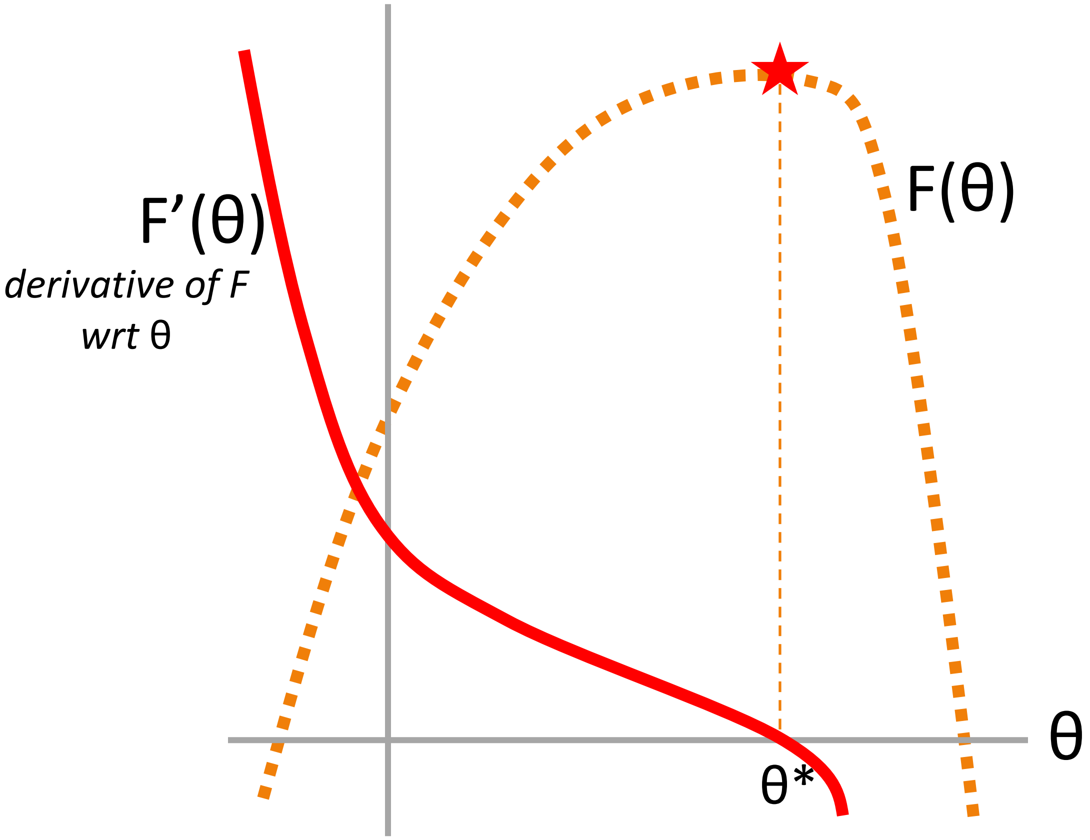


How will we optimize $F(\theta)$?

Calculus.


F(θ)
θ


F(θ)
θ
θ*


F’(θ)
derivative of F
wrt θ
F(θ)
θ
θ*

Example
(Best case, solve for roots of the derivative)

$$F(x) = -(x-2)^2$$

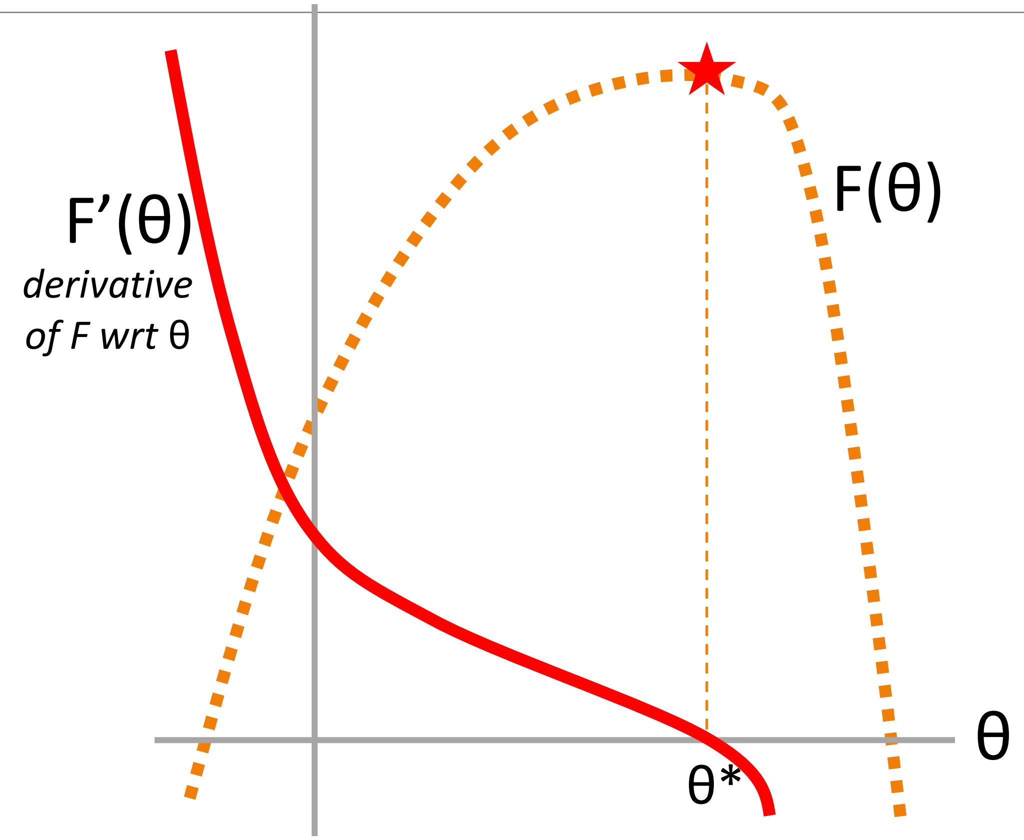
$$\Big\downarrow \textit{differentiate}$$

$$F'(x) = -2x + 4$$

$$\Big\downarrow \textit{Solve } F'(x) = 0$$

$$x = 2$$

What if you can’t find the roots? Follow the derivative



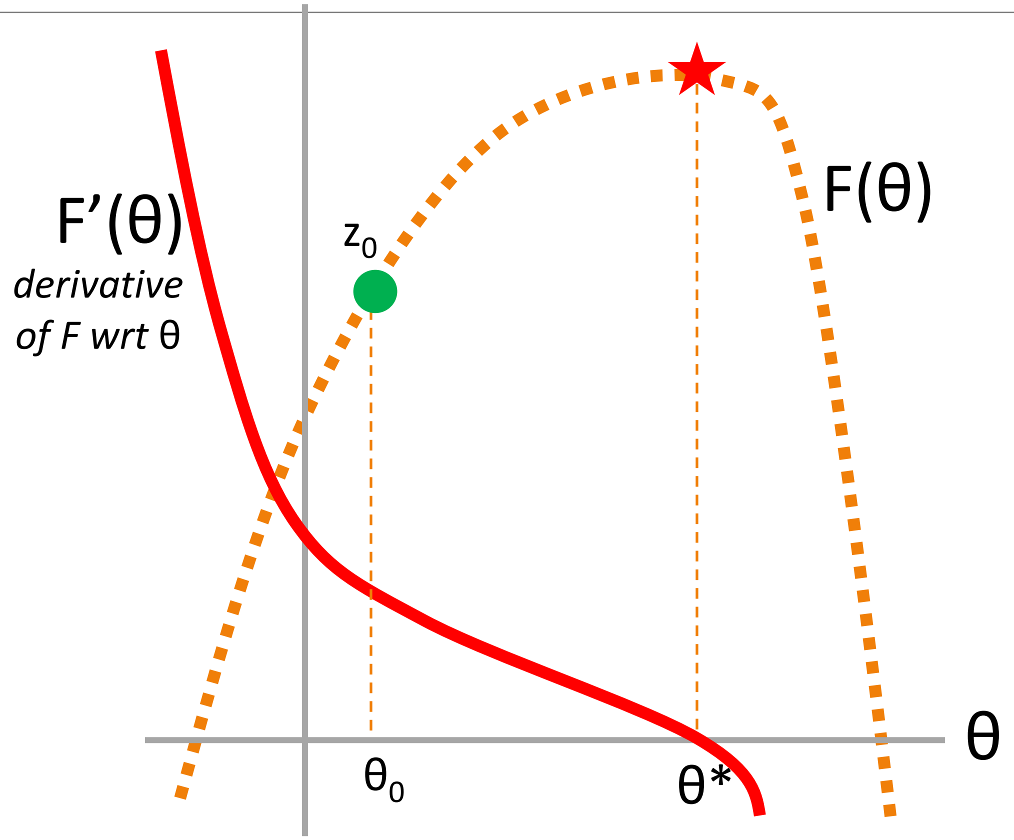
What if you can't find the roots? Follow the derivative

Set t = 0

Pick a starting value θ_t

Until converged:

1. Get value $z_t = F(\theta_t)$



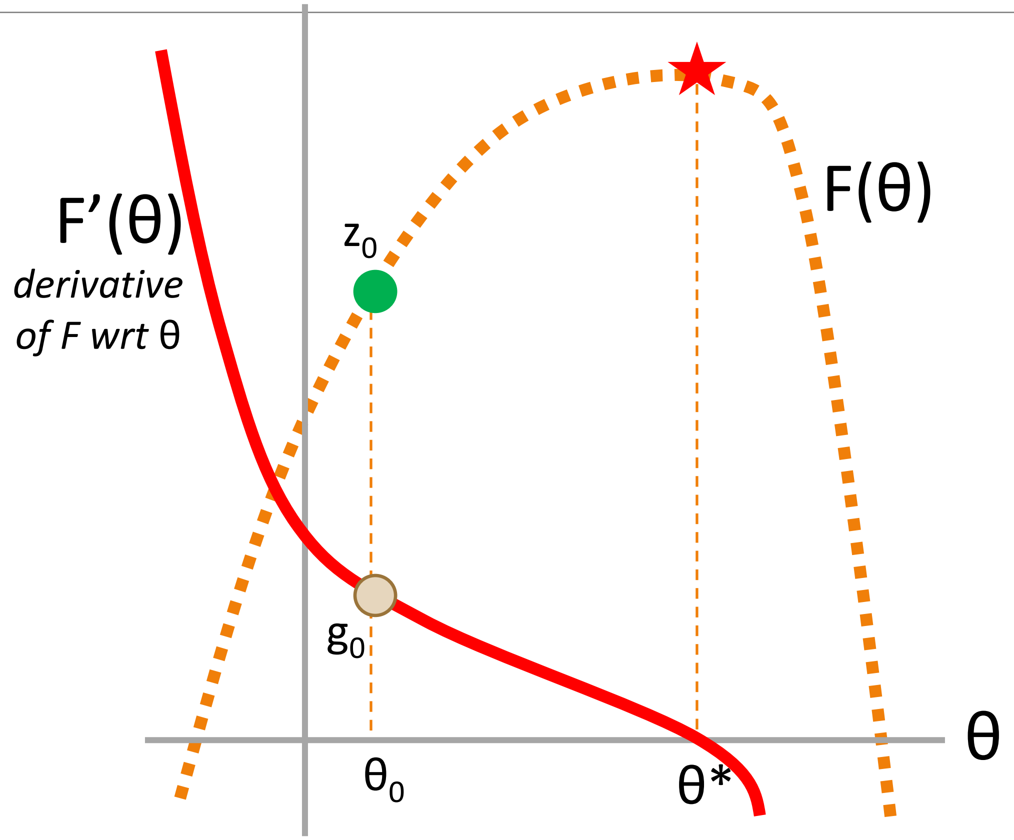
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $z_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$



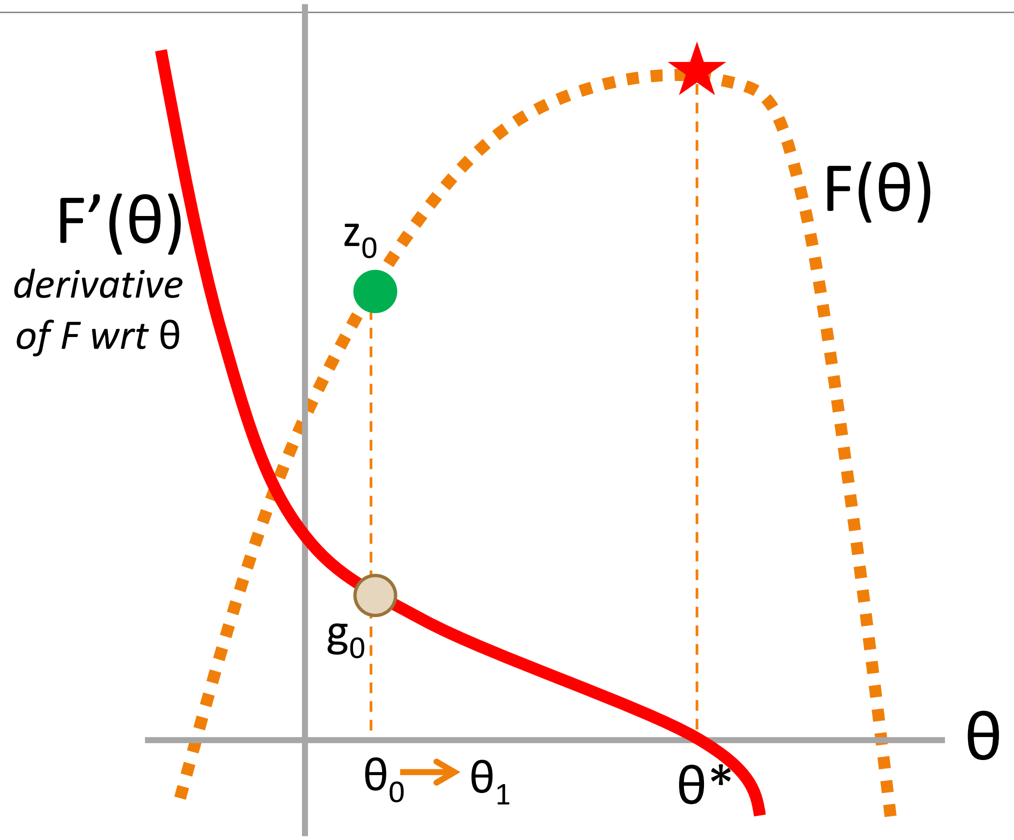
What if you can’t find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

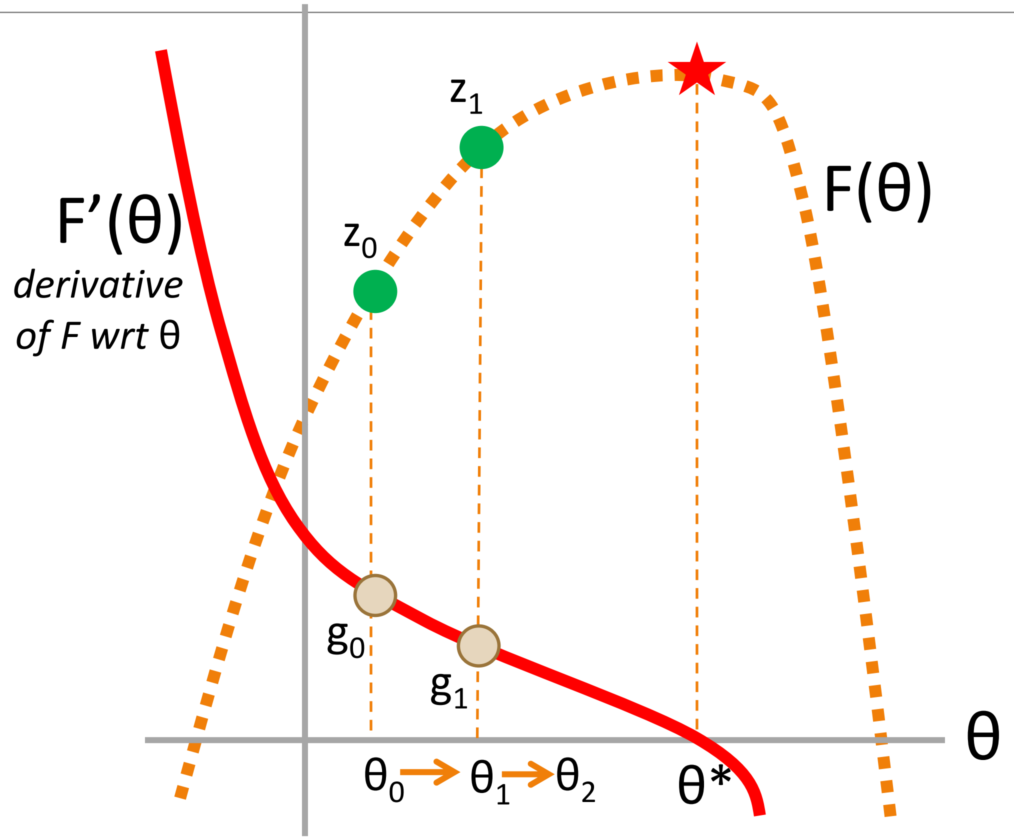
1. Get value $z_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



What if you can't find the roots? Follow the derivative

Set t = 0
Pick a starting value θ_t
Until converged:

1. Get value $z_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



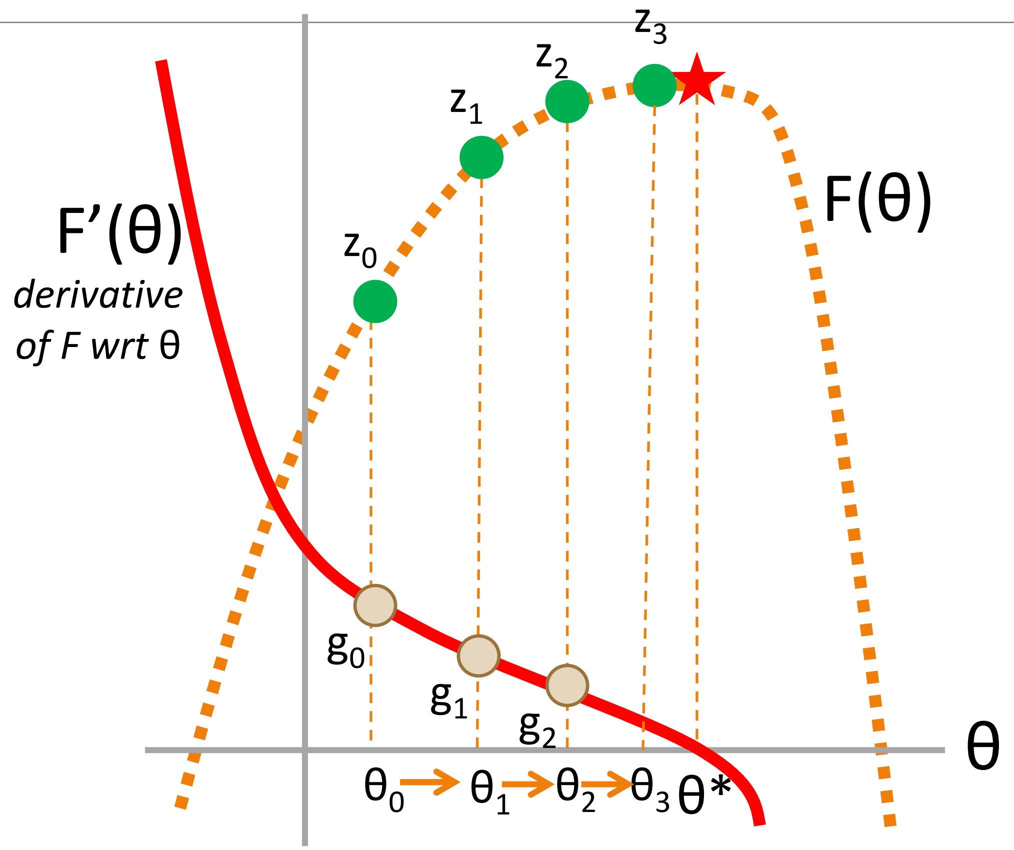
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $z_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1



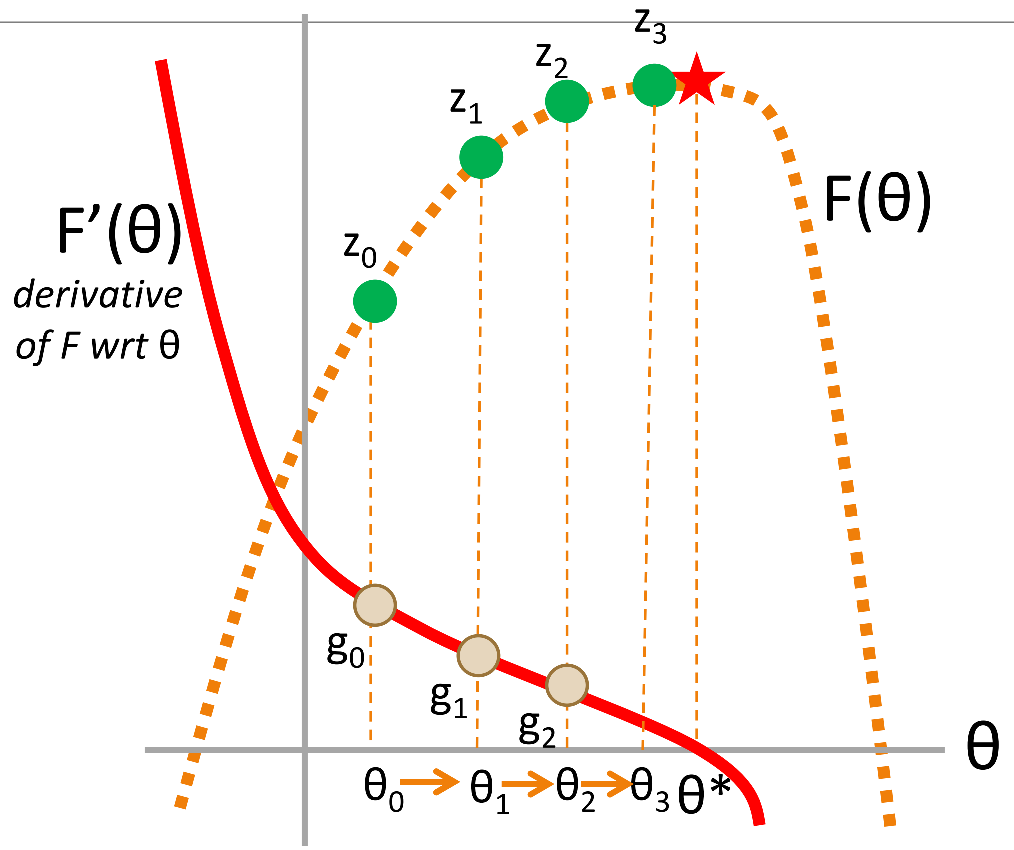
What if you can't find the roots? Follow the derivative

Set t = 0

Pick a starting value θ_t

Until **converged**:

1. Get value $z_t = F(\theta_t)$
2. Get derivative $g_t = F'(\theta_t)$
3. Get **scaling factor ρ_t**
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

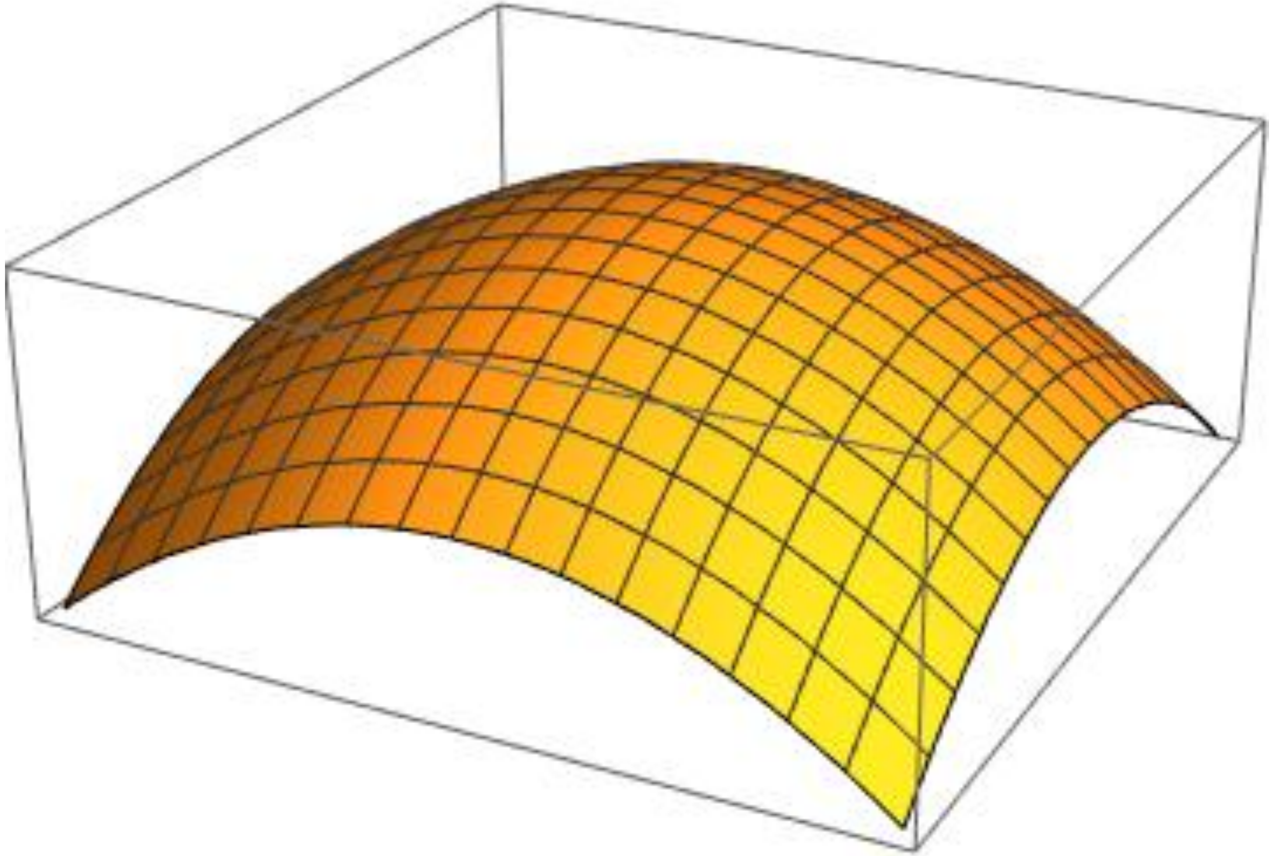


Gradient = Multi-variable derivative

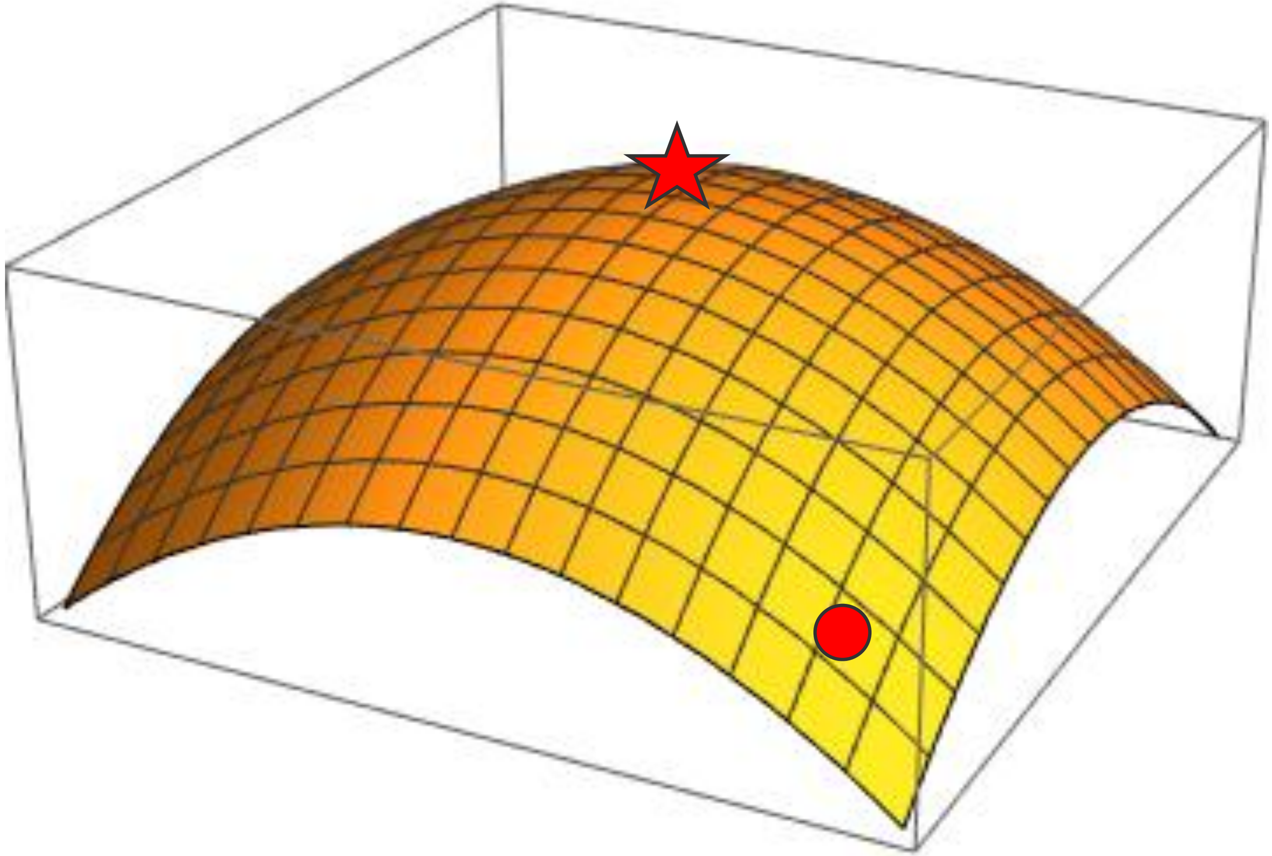
K-dimensional input

$$\nabla_\theta F(\theta) = \underbrace{\left(\frac{\partial F}{\partial \theta_1}, \frac{\partial F}{\partial \theta_2}, \ldots, \frac{\partial F}{\partial \theta_K}\right)}_{\text{K-dimensional output}}$$

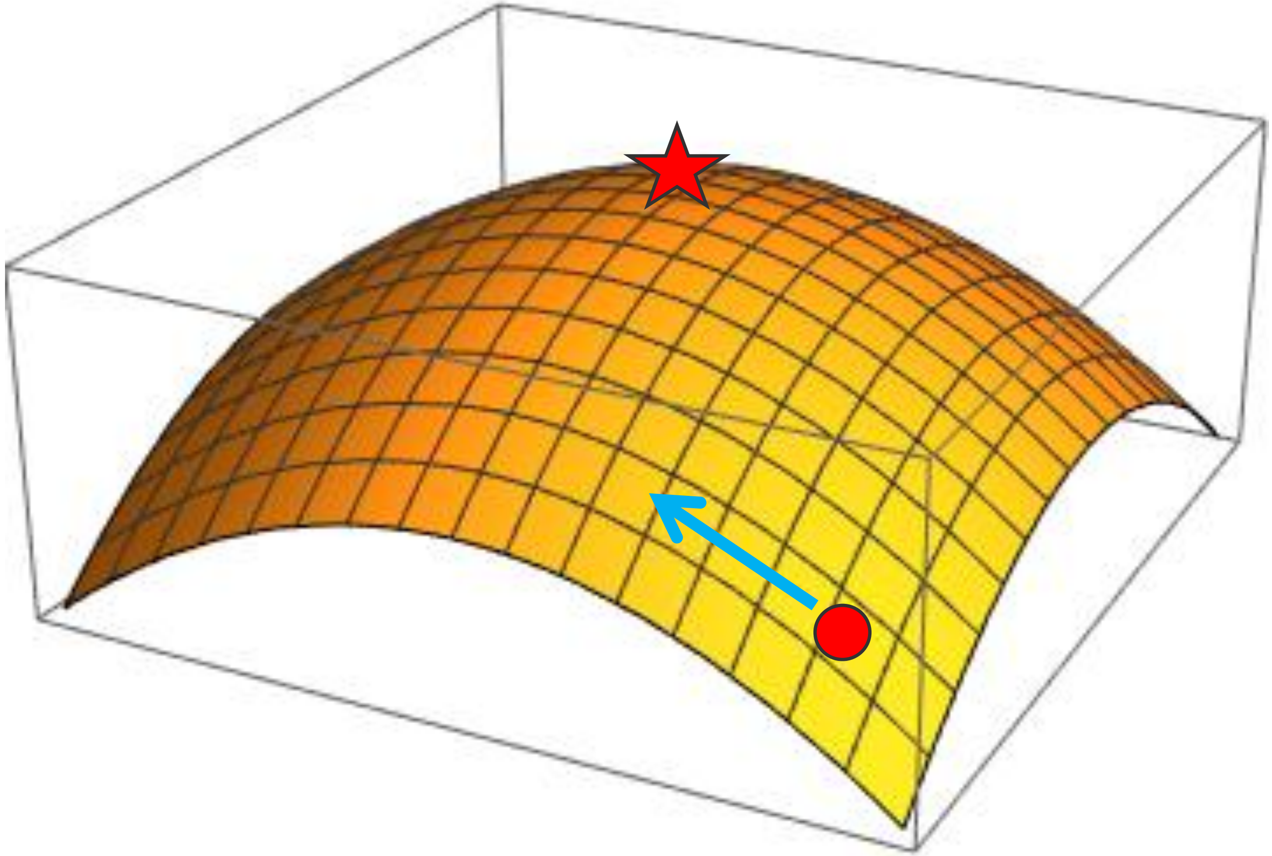
Gradient Ascent



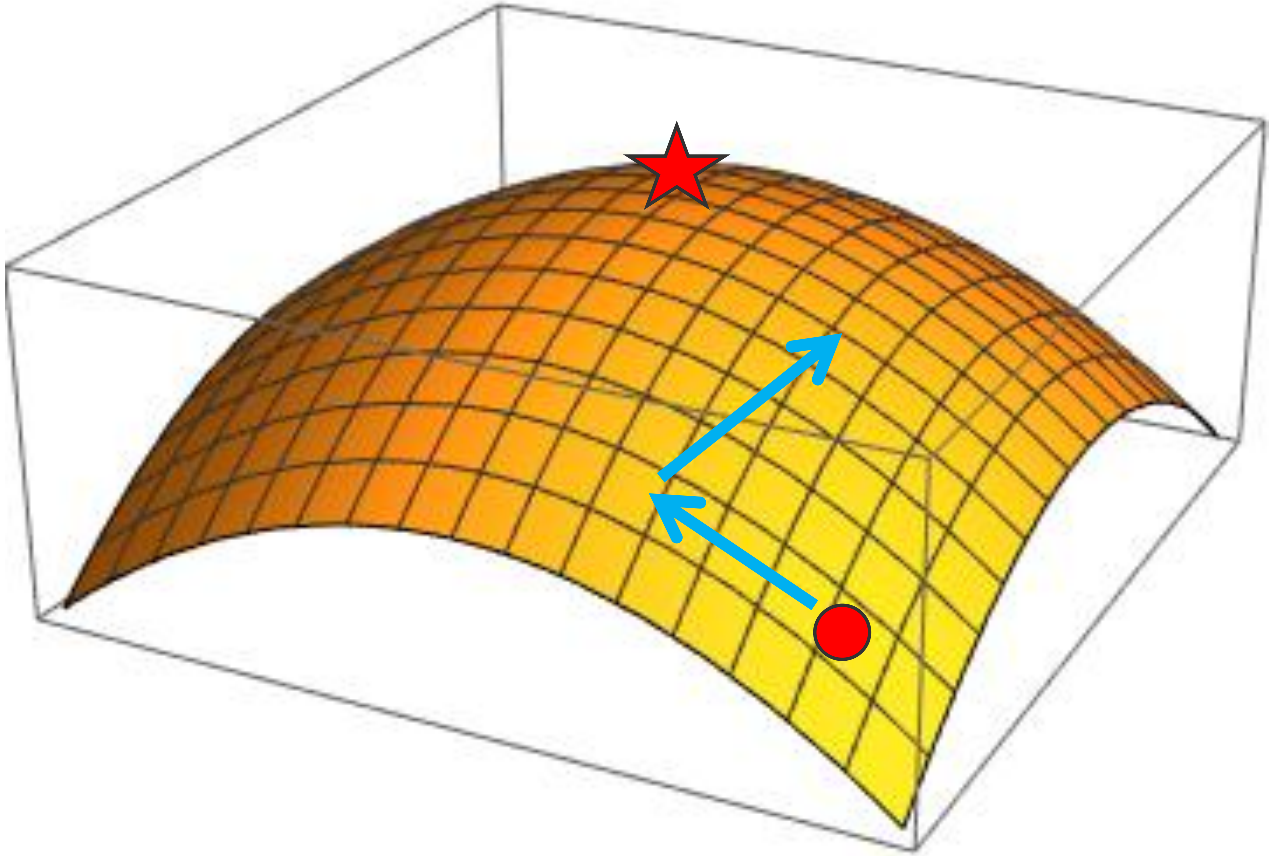
Gradient Ascent



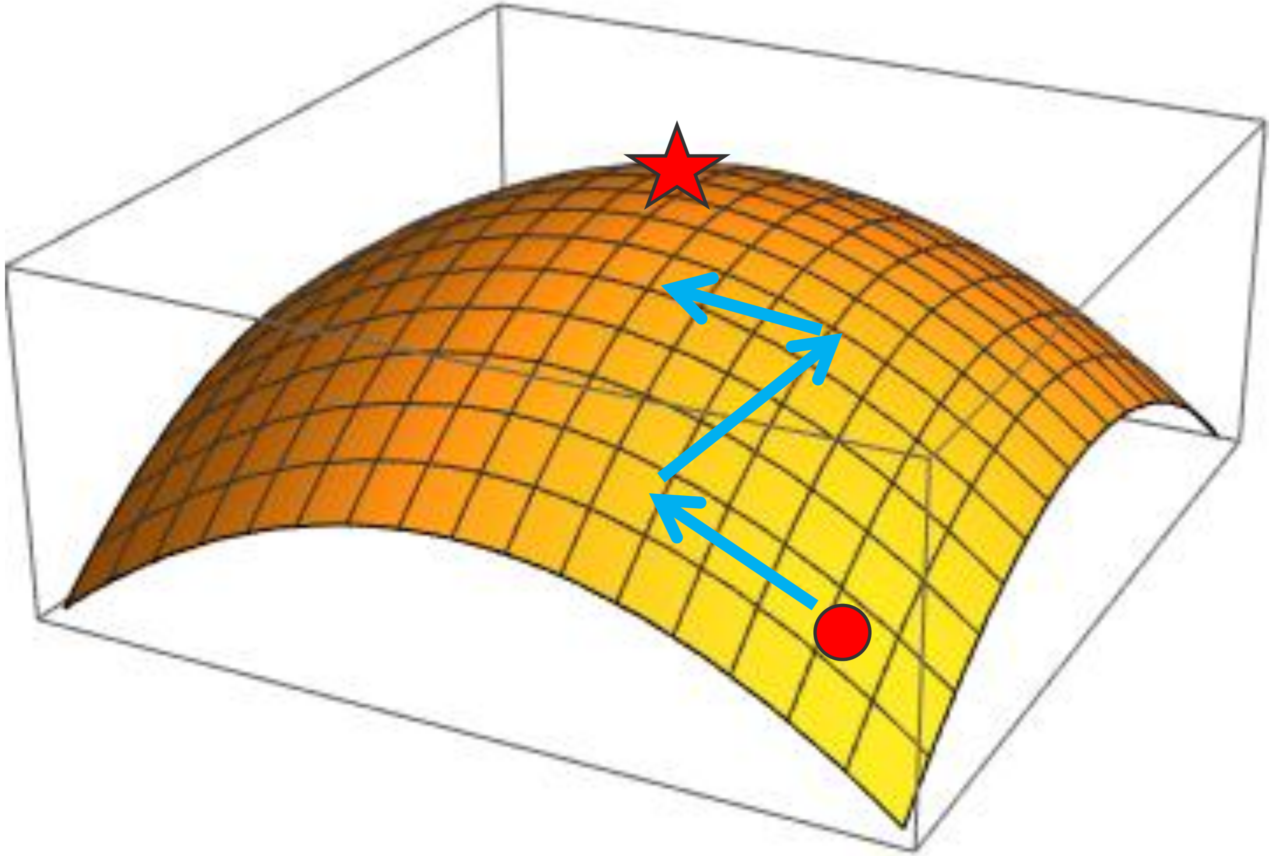
Gradient Ascent



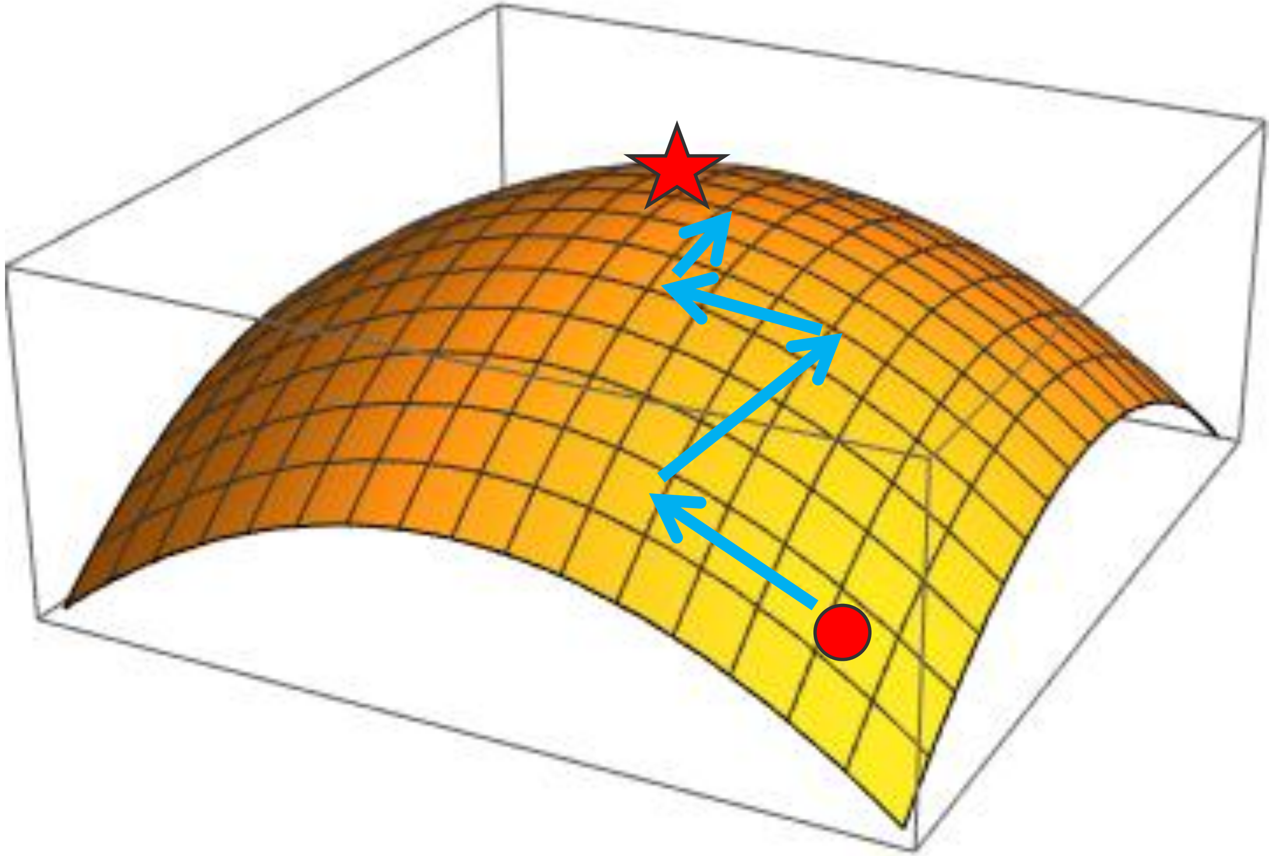
Gradient Ascent



Gradient Ascent



Gradient Ascent



What if you can't find the roots? Follow the **gradient**

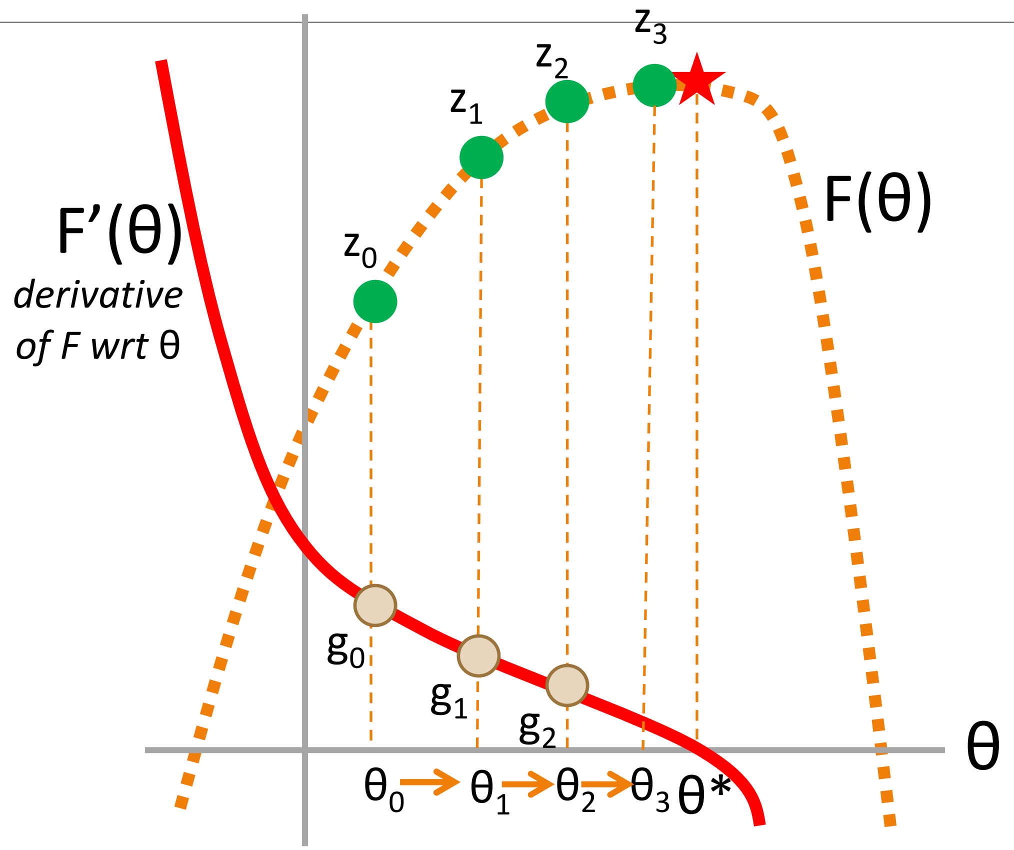
Set $t = 0$

Pick a starting value θ_t

Until converged:

1. Get value $z_t = F(\theta_t)$
2. Get **gradient** $g_t = F'(\theta_t)$
3. Get scaling factor ρ_t
4. Set $\theta_{t+1} = \theta_t + \rho_t * g_t$
5. Set t += 1

K-dimensional vectors



Outline

Maximum Entropy classifiers

- Defining the model: Discriminatively
- Defining the objective
- Learning: Optimizing the objective
- Defining the model: Generatively

Maxent Models for Classification: Discriminatively or Generatively Trained

Directly model the posterior

$$p(Y \mid X) = \mathbf{maxent}(X; Y)$$

Discriminatively trained classifier

Model the posterior with Bayes rule

$$p(Y \mid X) \propto \mathbf{maxent}(X \mid Y)p(Y)$$

Generatively trained classifier with maxent-based language model

Bayes' Rule

$$\underbrace{P(Y|X)}_{\text{Posterior}} = \frac{\overbrace{P(X|Y)}^{\text{Likelihood}} \cdot \overbrace{P(Y)}^{\text{Prior}}}{P(X)}$$

It's harder to model P(Y|X) directly since it might be that we only see that set of features once!

Bayes' Rule

$$P(c|d) = \frac{P(d|c)\cdot P(c)}{P(d)}$$

$$P\left(\textsc{Entailed}\ \middle|\ \boxed{\begin{array}{l}\text{s: Michael Jordan, coach Phil Jackson and the star}\\ \text{cast, including Scottie Pippen, took the Chicago}\\ \text{Bulls to six National Basketball Association}\\ \text{championships.}\\ \text{h: The Bulls basketball team is based in Chicago.}\end{array}}\right)$$

$$= \frac{P\left(\boxed{\begin{array}{l}\text{s: Michael Jordan, coach Phil Jackson and the star}\\ \text{cast, including Scottie Pippen, took the Chicago}\\ \text{Bulls to six National Basketball Association}\\ \text{championships.}\\ \text{h: The Bulls basketball team is based in Chicago.}\end{array}}\ \middle|\ \textsc{Entailed}\right)\cdot P\left(\textsc{Entailed}\right)}{P\left(\boxed{\begin{array}{l}\text{s: Michael Jordan, coach Phil Jackson and the star}\\ \text{cast, including Scottie Pippen, took the Chicago}\\ \text{Bulls to six National Basketball Association}\\ \text{championships.}\\ \text{h: The Bulls basketball team is based in Chicago.}\end{array}}\right)}$$

Bayes' Rule → Naïve Bayes Assumption

Bayes

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c) \cdot P(c)}{P(d)}$$

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c) \cdot P(c)}{\cancel{P(d)}}$$

We can make this assumption because P(d) stays the same regardless of the class!

Naïve Bayes

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) \approx \underset{c \in C}{\operatorname{argmax}} P(d|c) \cdot P(c)$$

Bayes' Rule → Naïve Bayes Assumption

Bayes

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c) \cdot P(c)}{P(d)}$$

Naïve Bayes

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) \approx \underset{c \in C}{\operatorname{argmax}} P(d|c) \cdot P(c)$$

Naïve bayes is **generative** because we are sort of assuming this is how the data point is generated: pick a class *c* and then generate the words by sampling from P(d|c)

SLP 4.1