

ML Evaluation + Logistic Regression Models

Instructor: Lara J. Martin (she/they)

TA: Omkar Kulkarni (he)

<https://laramartin.net/NLP-class/>

Slides modified from Dr. Frank Ferraro

Learning Objectives

Extend P/R to multi-class problems

Identify when you might want certain evaluation metrics over others

Model classification problems using logistic regression

Define appropriate features for a logistic regression problem

Review: Contingency Table


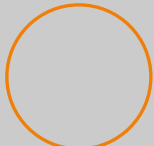

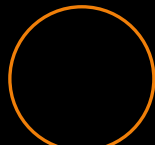
| <i>What label does our system predict? (↓)</i> | <i>What is the actual label?</i> | |
|--|-------------------------------------|----------------------------------|
| | Actual Target Class (“●”) | Not Target Class (“○”) |
| Selected/ Guessed (“●”) | | |
| Not selected/ not guessed (“○”) | | |

The Importance of “Polarity” in Binary Classification

What are you trying to “identify” in your classification?


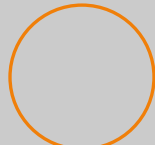



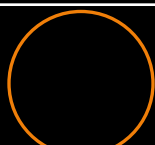


That is, are you trying to find ● or ○?

If  is our target







| | | Correct Value | |
|---------------|--|---|---|
| | |  |  |
| Guessed Value |  | ? | ? |
| |  | ? | ? |







Where do
TP / FP / FN / FN go?

If  is our target

| | | Correct Value | |
|---------------|--|--|--|
| | |  |  |
| Guessed Value |  | <i>TP</i>  | <i>FP</i>  |
| |  | <i>FN</i>  | <i>TN</i>  |

If  is our target


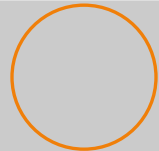



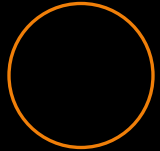


Predicted:      

Actual:      

$$acc = \frac{TP+TN}{TP+FP+FN+TN}$$

$$P = \frac{TP}{TP+FP}$$


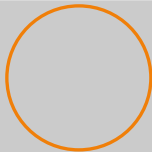

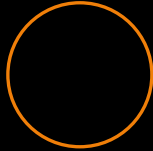
$$R = \frac{TP}{TP+FN}$$

| | | Correct Value | |
|---------------|--|---|---|
| | |  |  |
| Guessed Value |  | $TP = 2$  | $FP = 2$  |
| |  | $FN = 1$  | $TN = 1$  |

What are the
accuracy, recall, and
precision values?

Accuracy: 50%
Recall: 66.67%
Precision: 50%

If ○ is our target

| | | Correct Value | |
|---------------|---|---|---|
| | |  |  |
| Guessed Value |  | ? | ? |
| |  | ? | ? |

Where do
TP / FP / FN / FN go?

If ○ is our target

| | | Correct Value | |
|---------------|---|---------------|-------------|
| | | ● | ○ |
| Guessed Value | ● | <i>TN</i> ○ | <i>FN</i> ○ |
| | ○ | <i>FP</i> ○ | <i>TP</i> ○ |

If ○ is our target

Predicted: ○ ● ● ● ○ ●

Actual: ● ● ● ○ ○ ○

$$acc = \frac{TP+TN}{TP+FP+FN+TN}$$

$$P = \frac{TP}{TP+FP}$$


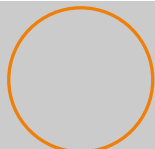

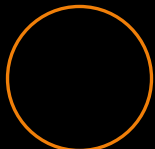
$$R = \frac{TP}{TP+FN}$$

| | | Correct Value | |
|---------------|---|------------------|------------------|
| | | ● | ○ |
| Guessed Value | ● | $TN \quad ○ = 2$ | $FN \quad ○ = 2$ |
| | ○ | $FP \quad ○ = 1$ | $TP \quad ○ = 1$ |

What are the
accuracy, recall, and
precision values?

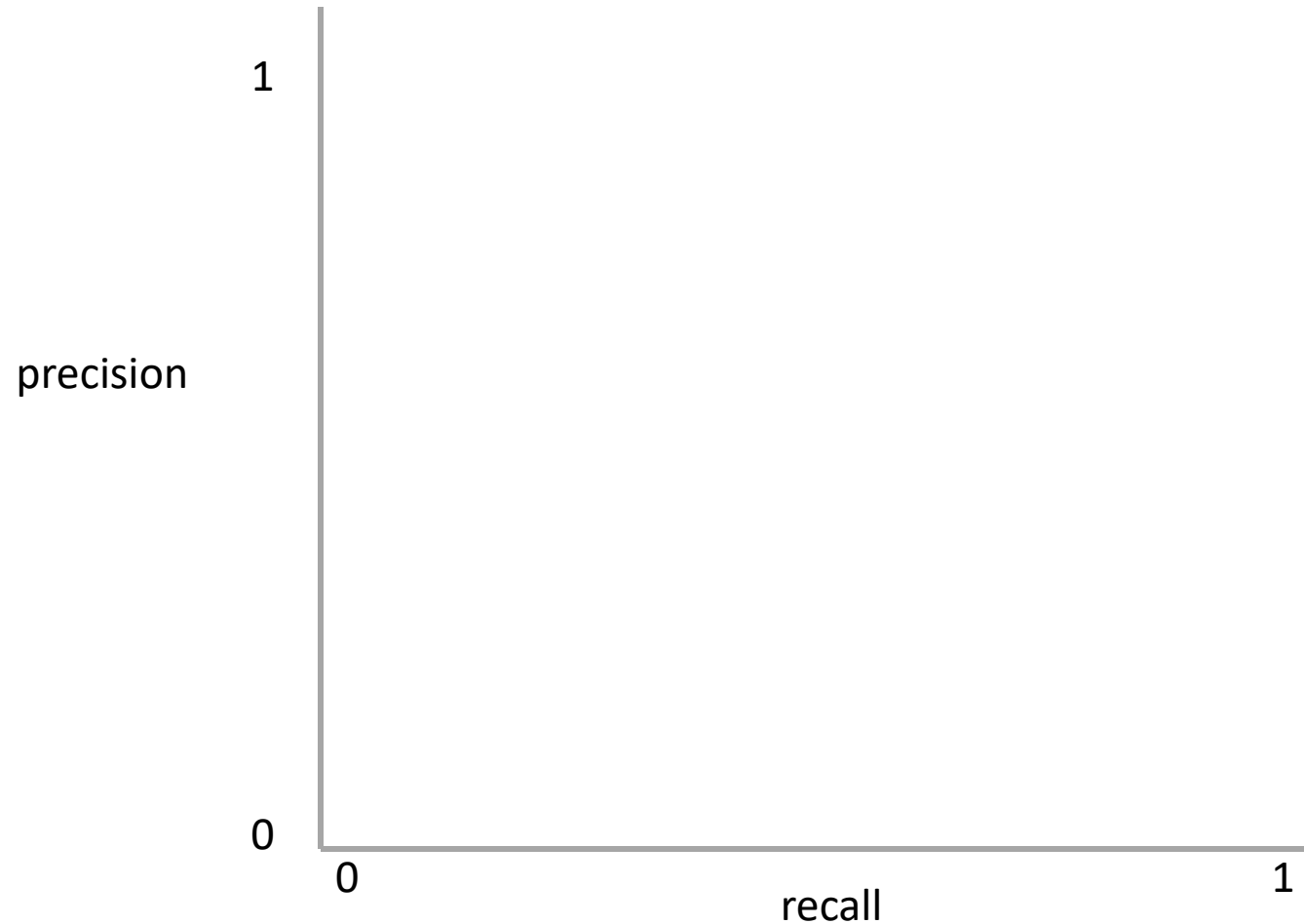
Accuracy: 50%
Recall: 33.34%
Precision: 50%

When there are two classes, TP/TN & FP/FN are symmetrical

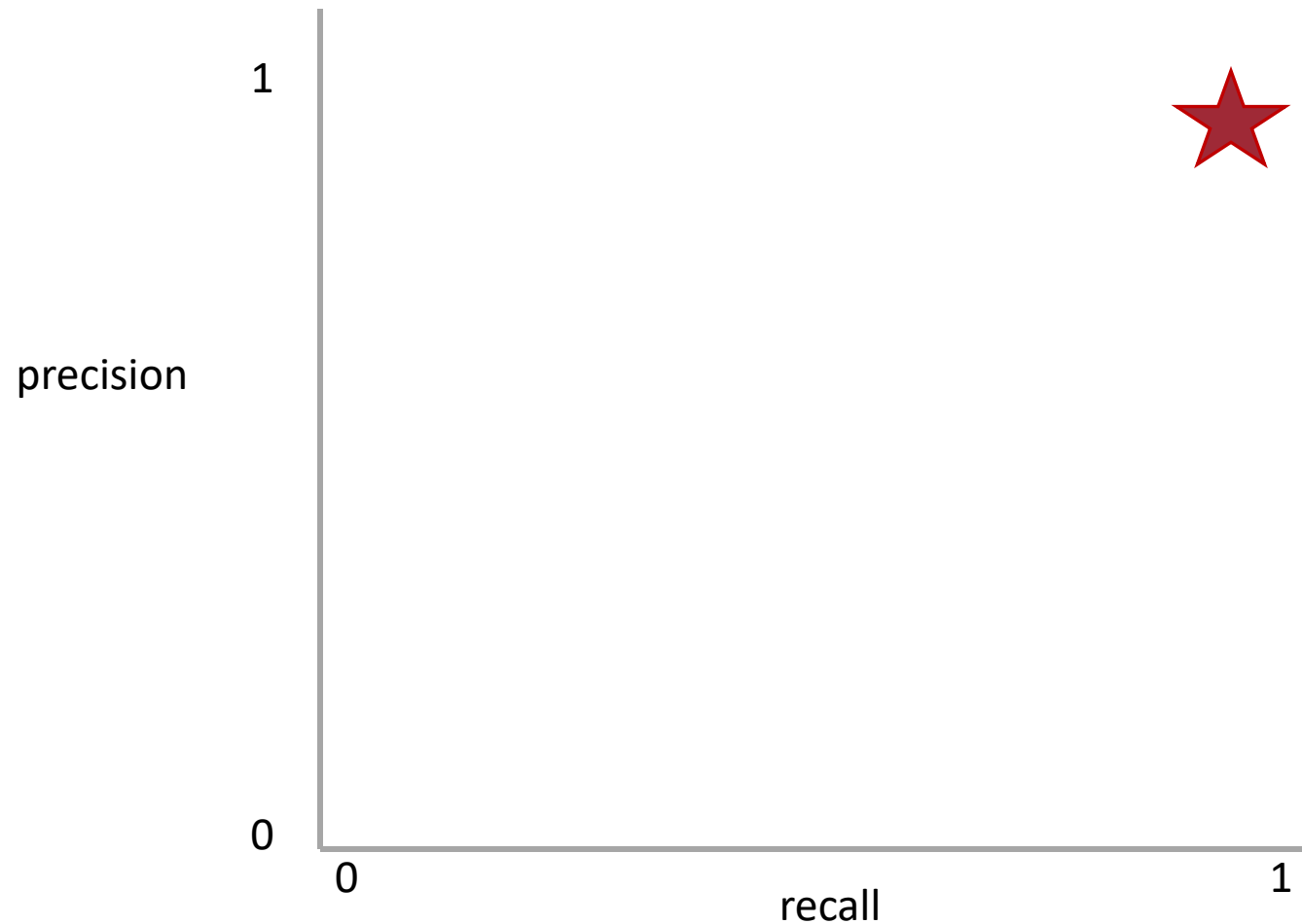
| | | Correct Value | |
|---------------|---|---|---|
| | |  |  |
| Guessed Value |  | $TP \text{ } = TN \text{ } $ | $FP \text{ } = FN \text{ } $ |
| |  | $FN \text{ } = FP \text{ } $ | $TN \text{ } = TP \text{ } $ |

Precision and Recall Present a Tradeoff

Q: Where do you
want your ideal
model ?



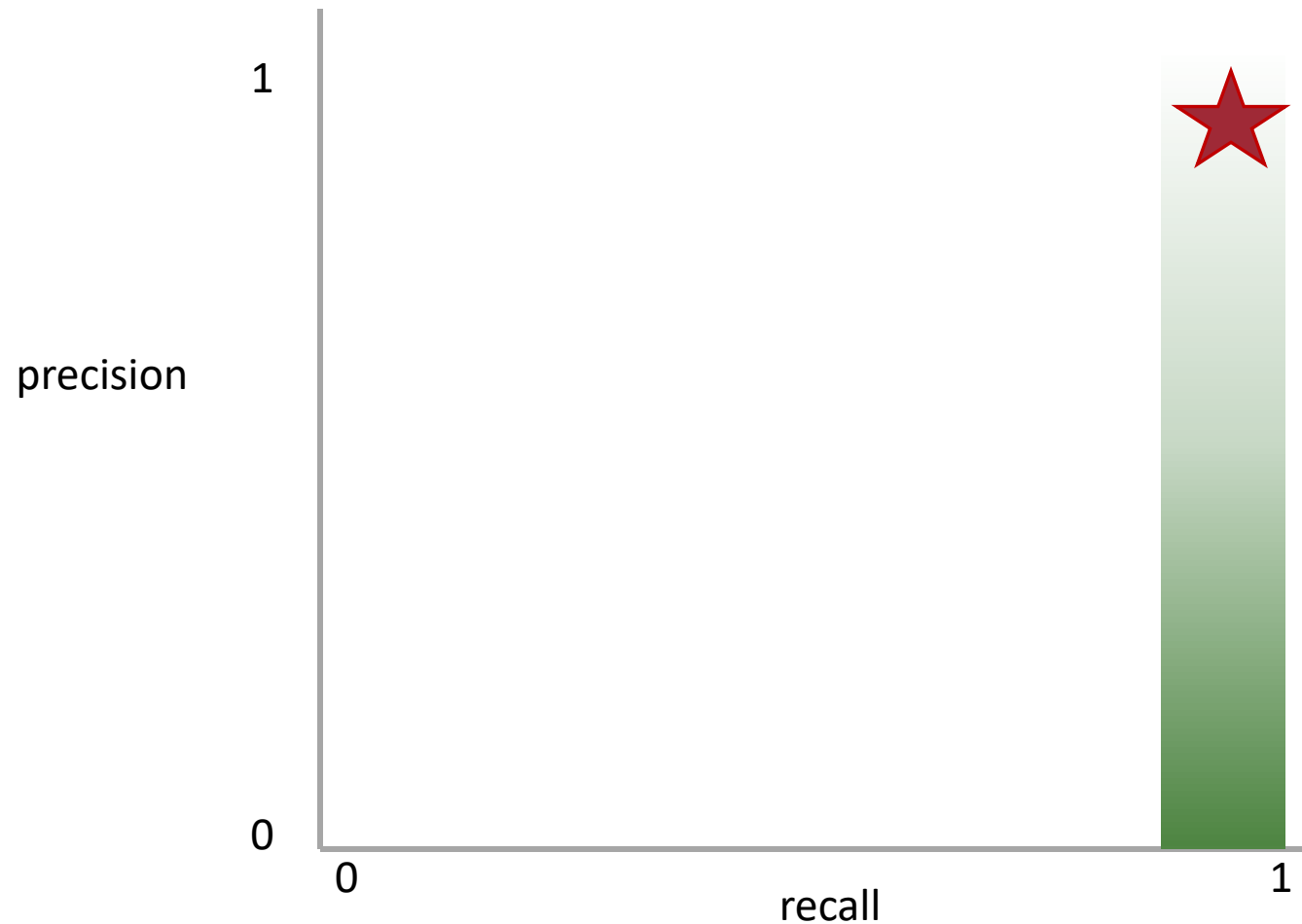
Precision and Recall Present a Tradeoff



Q: Where do you want your ideal model?

Q: You have a model that always identifies correct instances. Where on this graph is it?

Precision and Recall Present a Tradeoff

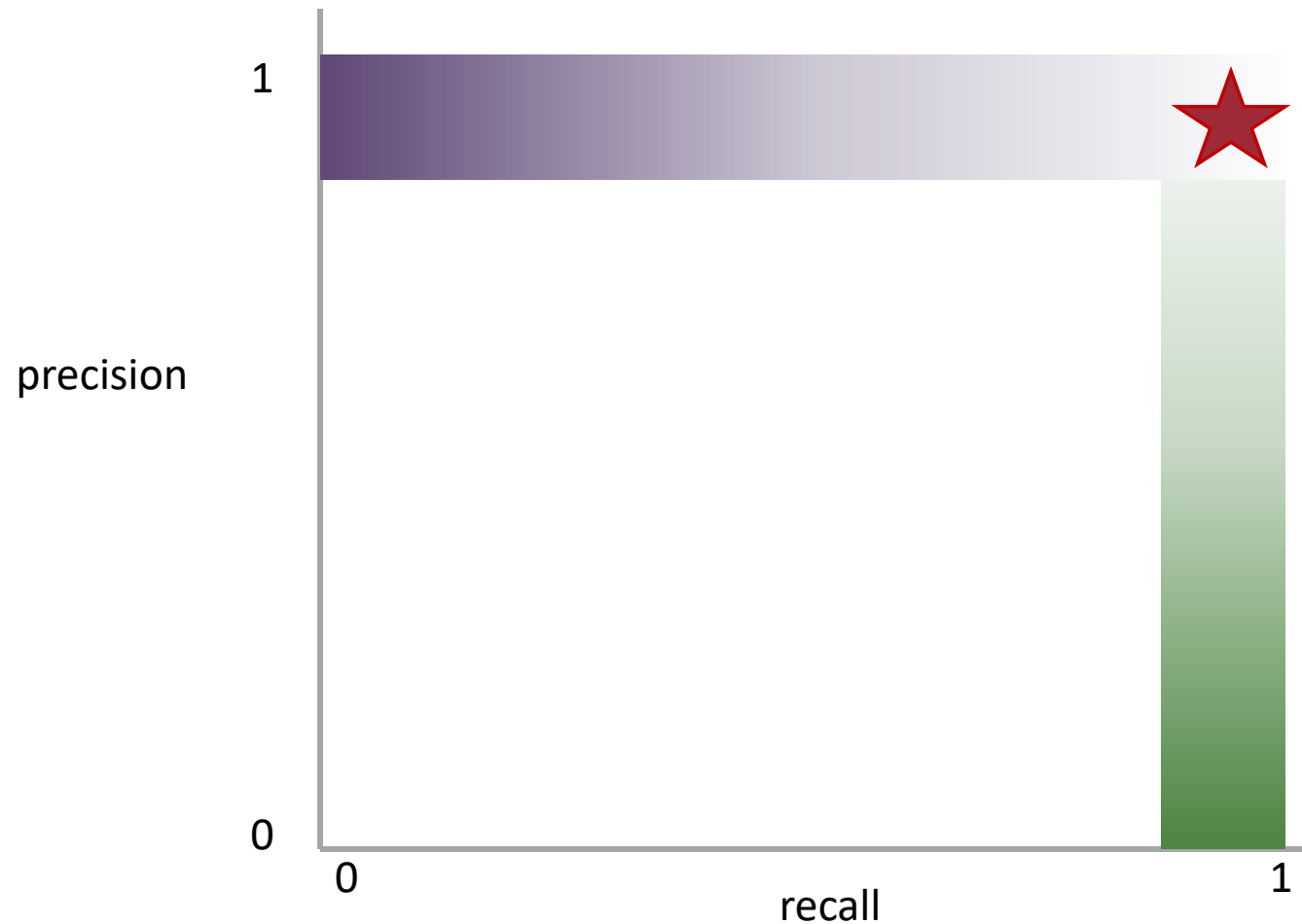


Q: Where do you want your ideal **model** ?

Q: You have a **model** that always identifies correct instances. Where on this graph is it?

Q: You have a **model** that only make correct predictions. Where on this graph is it?

Precision and Recall Present a Tradeoff

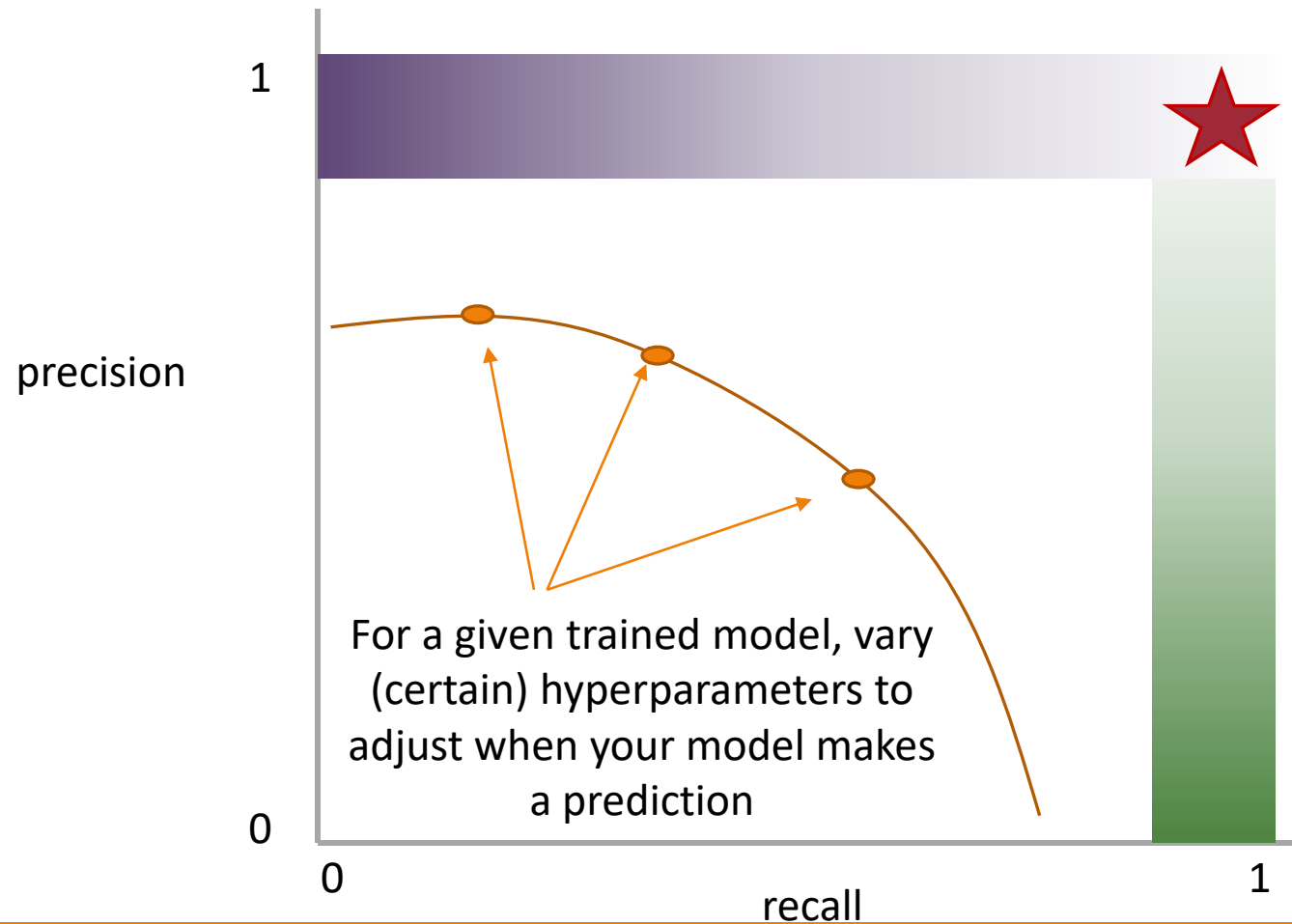


Q: Where do you want your ideal model ?

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only make correct predictions. Where on this graph is it?

Precision and Recall Present a Tradeoff



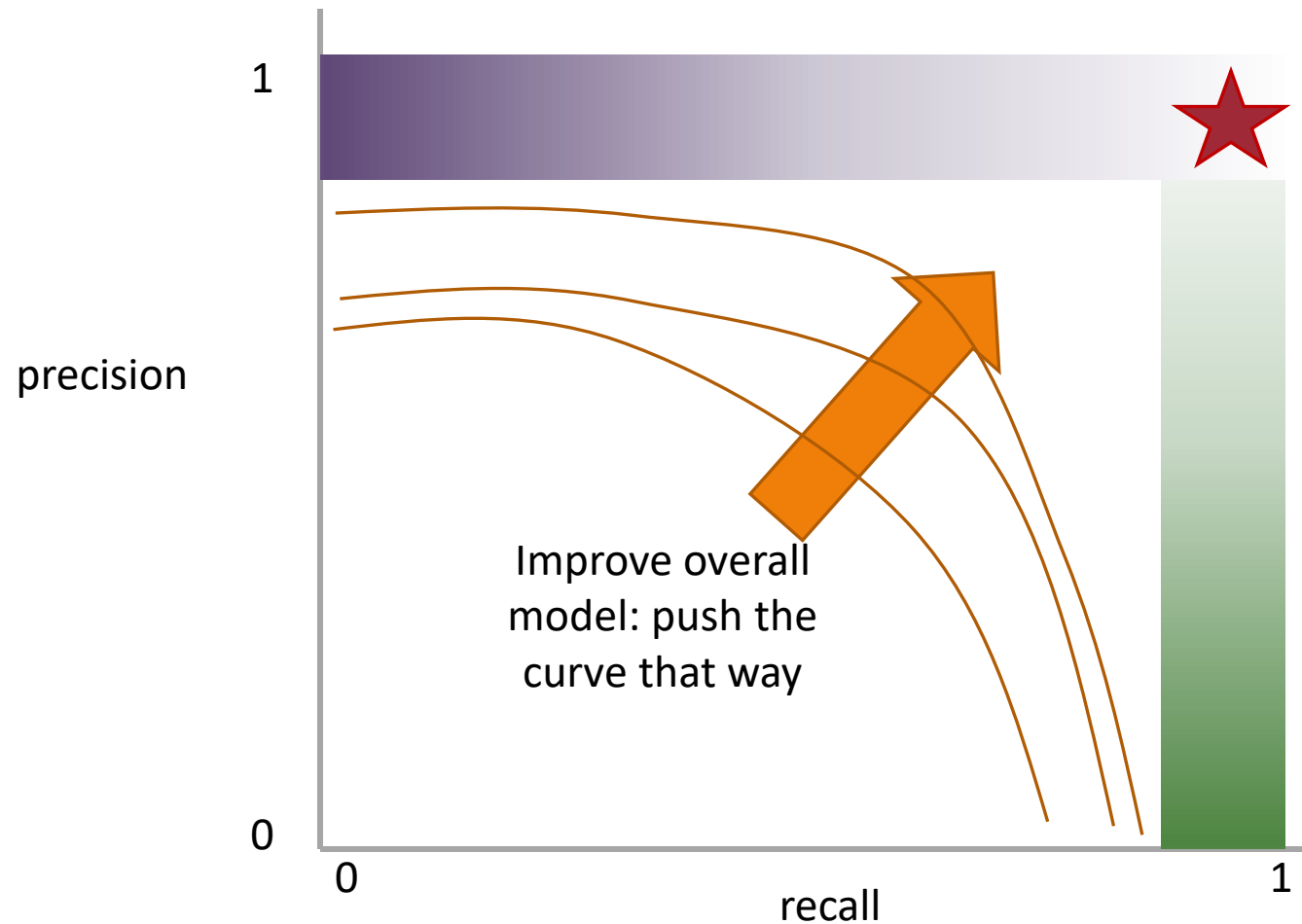
Q: Where do you want your ideal model ?

Q: You have a model that always identifies correct instances. Where on this graph is it?

Q: You have a model that only make correct predictions. Where on this graph is it?

Idea: measure the tradeoff between precision and recall

Precision and Recall Present a Tradeoff



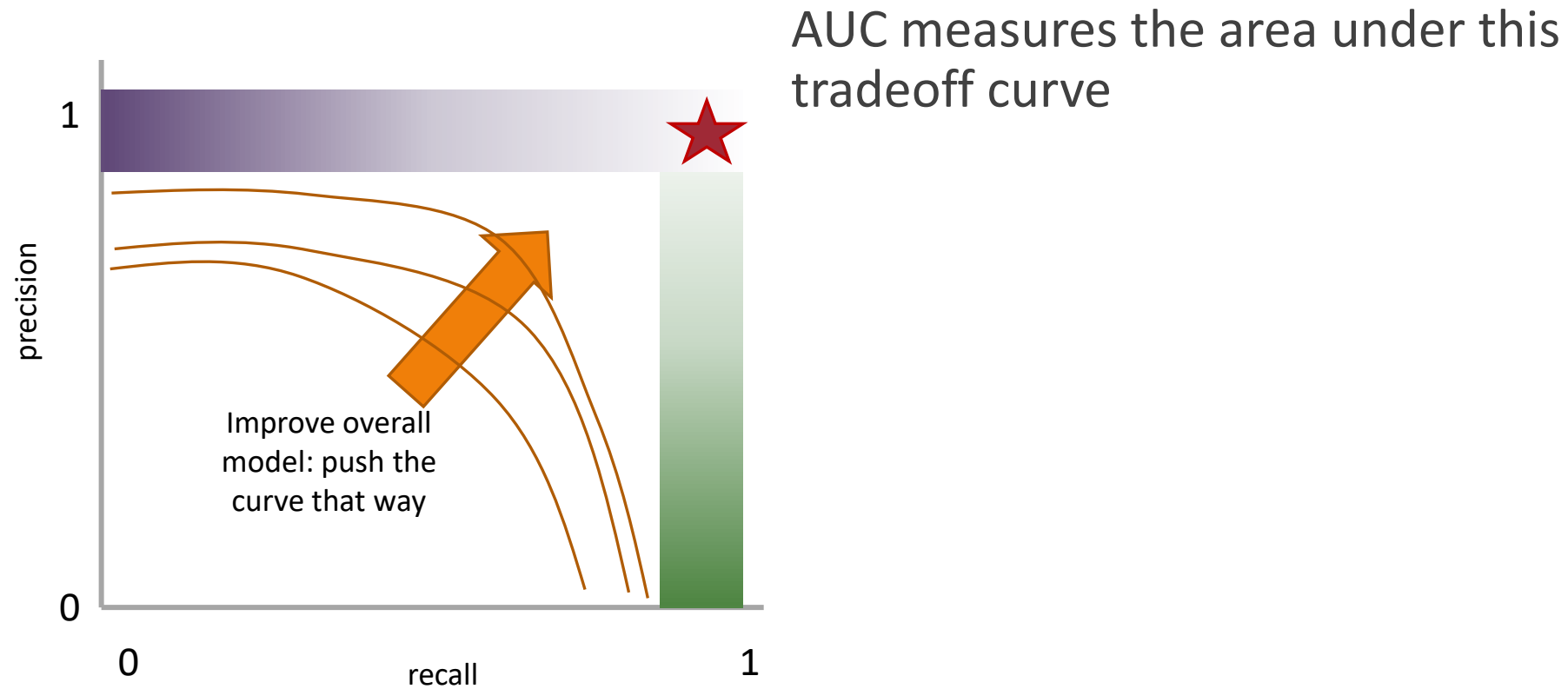
Q: Where do you want your ideal model ?

Q: You have a model that always identifies correct instances. Where on this graph is it?

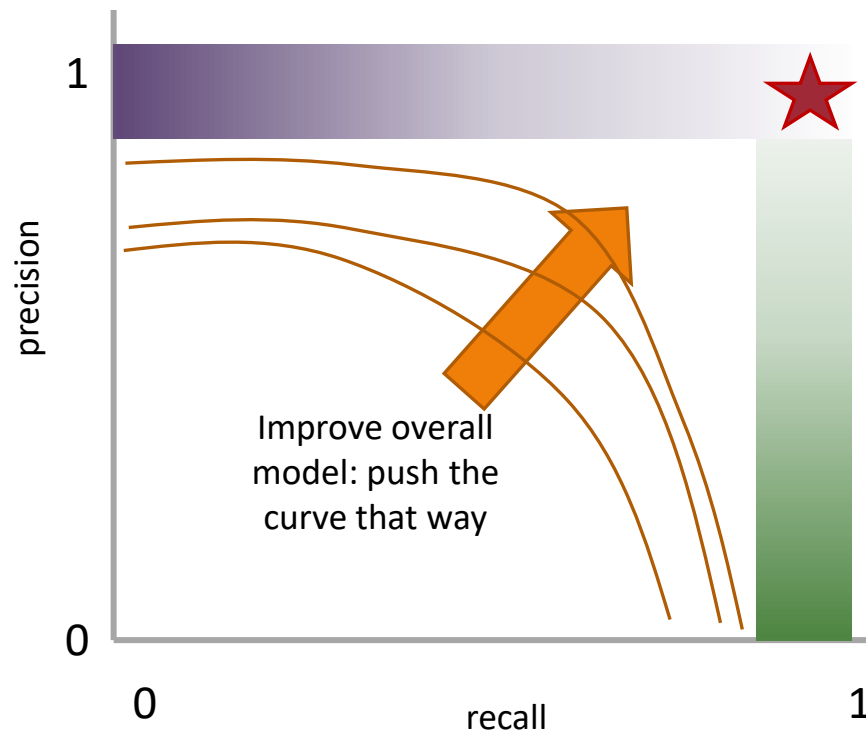
Q: You have a model that only make correct predictions. Where on this graph is it?

Idea: measure the tradeoff between precision and recall

Measure this Tradeoff: Area Under the Curve (AUC)



Measure this Tradeoff: Area Under the Curve (AUC)



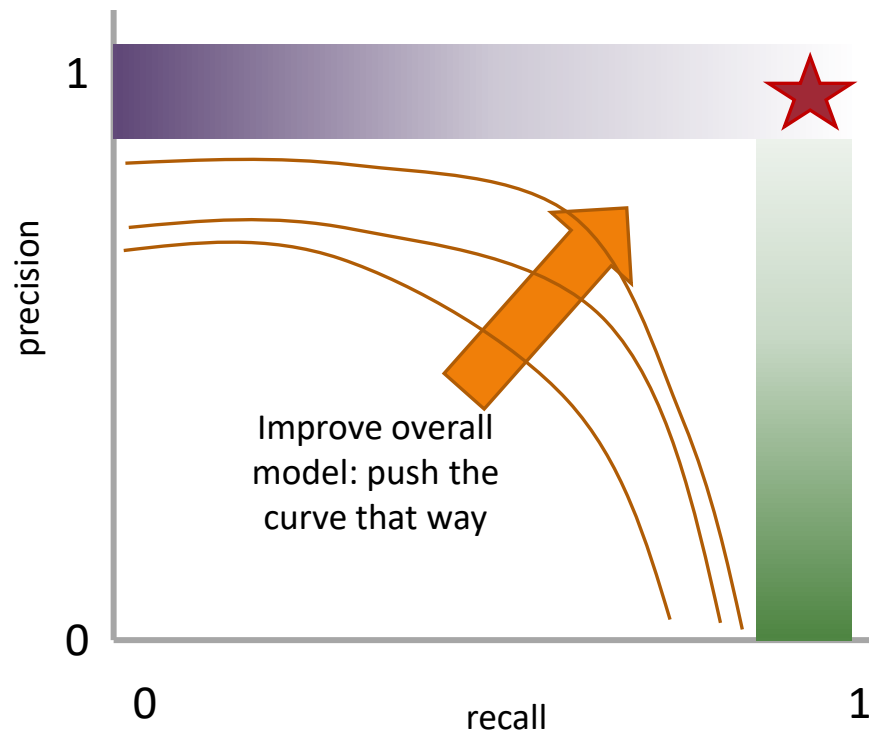
AUC measures the area under this tradeoff curve

1. Computing the curve

You need true labels & predicted labels with some score/confidence estimate

Threshold the scores and for each threshold compute precision and recall

Measure this Tradeoff: Area Under the Curve (AUC)



AUC measures the area under this tradeoff curve

1. Computing the curve
You need true labels & predicted labels with some score/confidence estimate
Threshold the scores and for each threshold compute precision and recall
2. Finding the area
How to implement: trapezoidal rule (& others)

In practice: external library like the `sklearn.metrics` module

A combined measure: F1 (or F-score)

Weighted (harmonic) average of **P**recision & **R**ecall

F1 measure: equal weighting between precision and recall

$$F_1 = \frac{2 * P * R}{P + R}$$

A combined measure: F1 (or F-score)

Weighted (harmonic) average of **P**recision & **R**ecall

F1 measure: equal weighting between precision and recall

$$F_1 = \frac{2 * P * R}{P + R} = \frac{2 * TP}{2 * TP + FP + FN}$$

(useful when $P = R = 0$)

Comparing Accuracy & F1

Accuracy: % of items correct

$$\frac{TP + TN}{TP + FP + FN + TN}$$

$$F_1 = \frac{2 * P * R}{P + R} = \frac{2 * TP}{2 * TP + FP + FN}$$

When would you want to use accuracy vs F1?

Accuracy works better if the dataset is balanced

Accuracy takes everything in consideration

F-Score is focused on TP

| | Actually Target | Actually Not Target |
|------------------------|---------------------|---------------------|
| Selected/Guessed | True Positive (TP) | False Positive (FP) |
| Not select/not guessed | False Negative (FN) | True Negative (TN) |

P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

If we have more than one class, how do we combine multiple performance measures into one quantity?

P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

Macroaveraging: Compute performance for each class, then average.

$$\text{macroprecision} = \frac{1}{C} \sum_c \frac{TP_c}{TP_c + FP_c} = \frac{1}{C} \sum_c \text{precision}_c$$

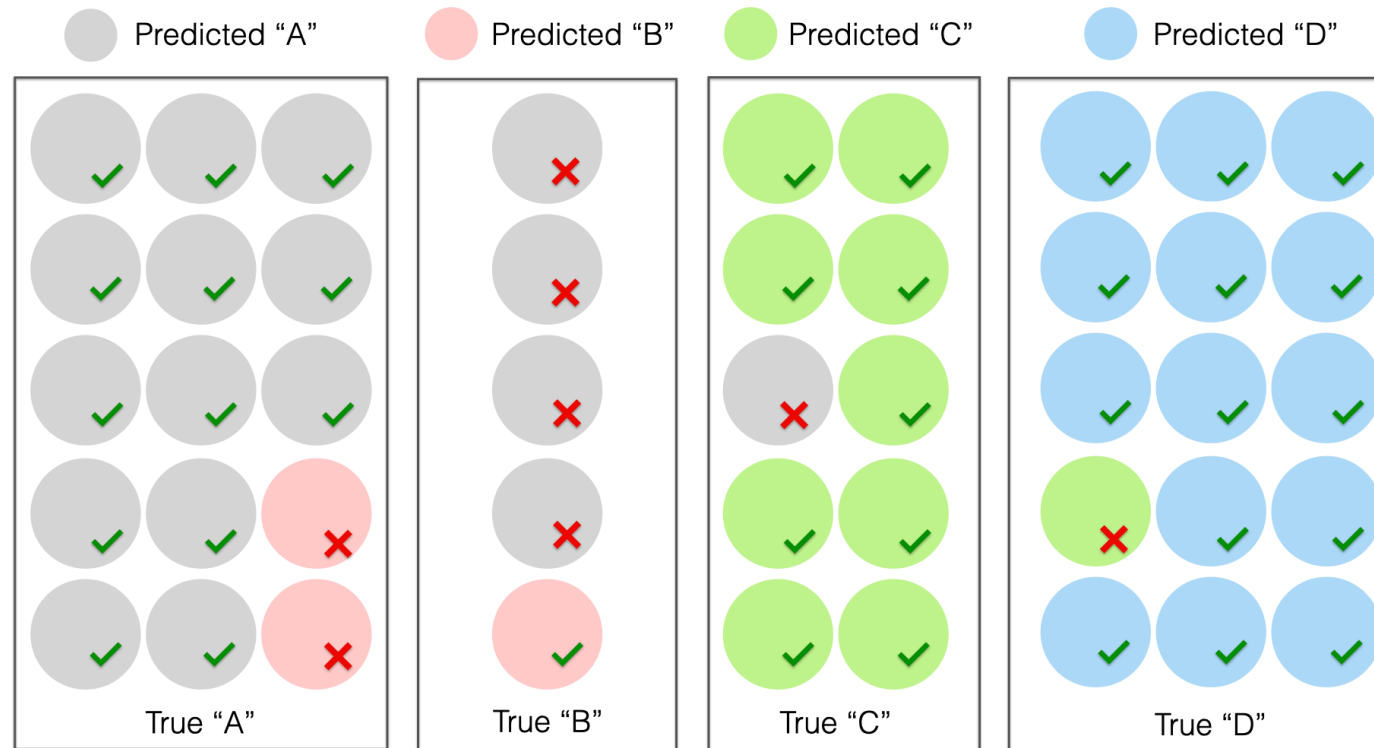
$$\text{macrorecall} = \frac{1}{C} \sum_c \frac{TP_c}{TP_c + FN_c} = \frac{1}{C} \sum_c \text{recall}_c$$

Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.

$$\text{microprecision} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FP_c}$$

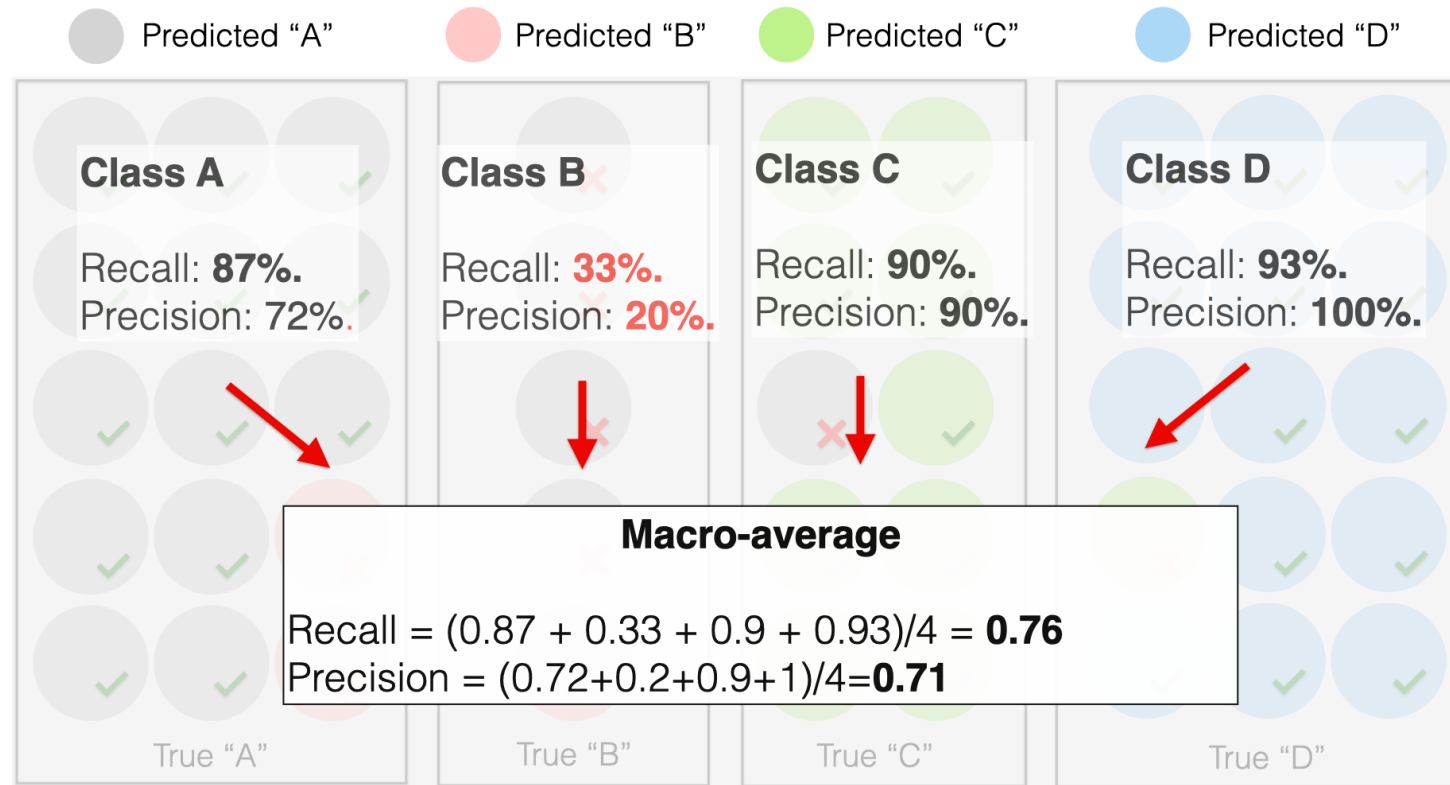
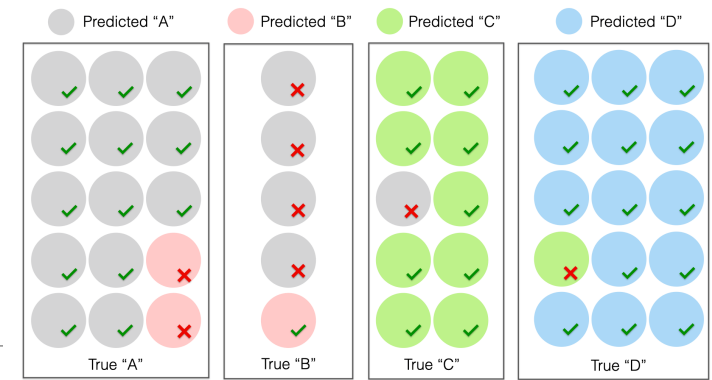
$$\text{microrecall} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FN_c}$$

Macro/Micro Example



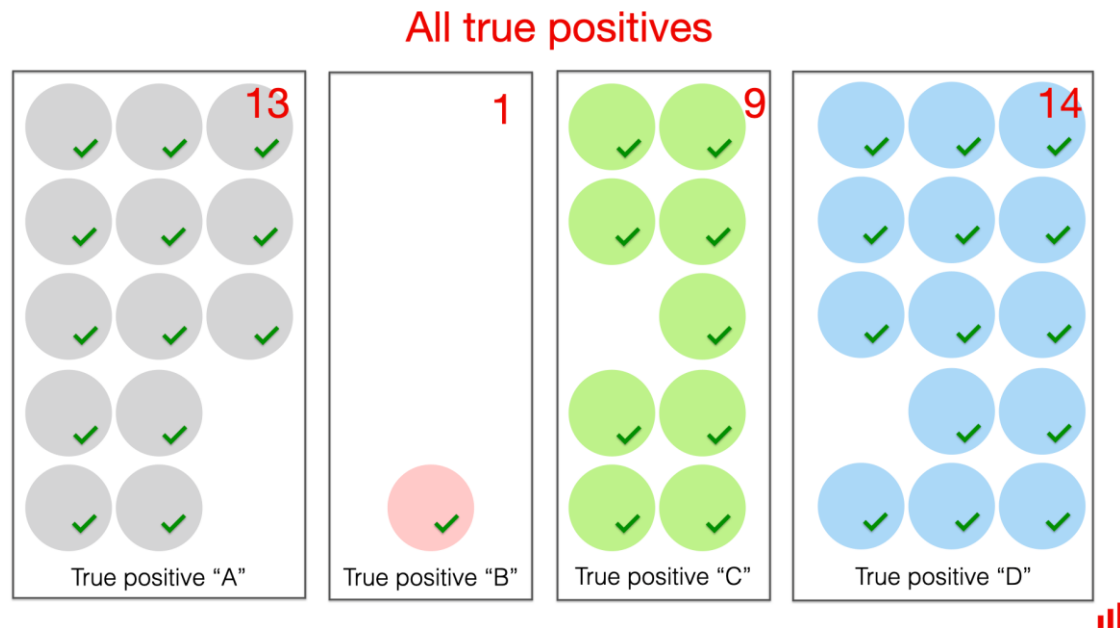
Each *class* has equal weight

Macro-Average



Each *instance* has equal weight

Micro-Average



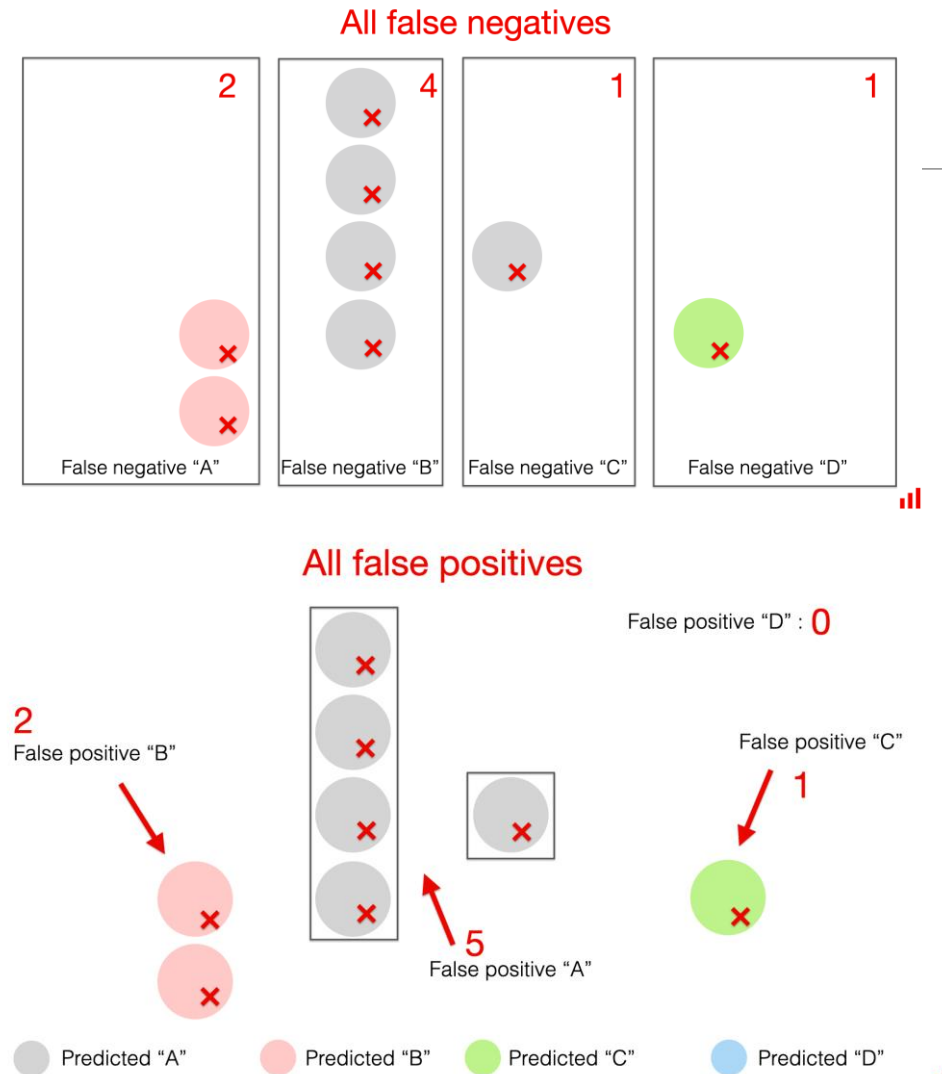
Total TP: 13 + 1 + 9 + 14 = 37

Total FP: 2 + 5 + 1 + 0 = 8

Total FN: 2 + 4 + 1 + 1 = 8

$$\text{Precision}_{\text{Micro-average}} = \frac{13 + 1 + 9 + 14}{13 + 1 + 9 + 14 + 2 + 5 + 1 + 0} = 0.82$$
$$\text{Recall}_{\text{Micro-average}} = \frac{13 + 1 + 9 + 14}{13 + 1 + 9 + 14 + 2 + 4 + 1 + 1} = 0.82$$

<https://www.evidentlyai.com/classification-metrics/multi-class-metrics>



P/R/F in a Multi-class Setting: Micro- vs. Macro-Averaging

Macroaveraging: Compute performance for each class, then average.

$$\text{macroprecision} = \frac{1}{C} \sum_c \frac{TP_c}{TP_c + FP_c} = \frac{1}{C} \sum_c \text{precision}_c$$

$$\text{macrorecall} = \frac{1}{C} \sum_c \frac{TP_c}{TP_c + FN_c} = \frac{1}{C} \sum_c \text{recall}_c$$

Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.

$$\text{microprecision} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FP_c}$$

$$\text{microrecall} = \frac{\sum_c TP_c}{\sum_c TP_c + \sum_c FN_c}$$

When would we want to prefer micro-averaging vs macro-averaging?

But how do we compute stats for multiple classes?

We already saw how the “polarity” affects the stats we compute...

Two main approaches. Either:

1. Compute “one-vs-all” 2x2 tables. OR
2. Generalize the 2x2 tables and compute per-class TP / FP / FN based on the diagonals and off-diagonals



1. Compute “one-vs-all” 2x2 tables


Predicted



Actual



| Look for  | Actually Target | Actually Not Target | Look for  | Actually Target | Actually Not Target |
|---|---------------------------|--------------------------|---|---------------------------|--------------------------|
| Selected/G uessed | True Positive (TP) | False Positive (FP) | Selected/G uessed | True Positive (TP) | False Positive (FP) |
| Not select/not guessed | False Negative (FN) | True Negative (TN) | Not select/not guessed | False Negative (FN) | True Negative (TN) |

| Look for  | Actually Target | Actually Not Target |
|--|---------------------------|--------------------------|
| Selected/G uessed | True Positive (TP) | False Positive (FP) |
| Not select/not guessed | False Negative (FN) | True Negative (TN) |

1. Compute “one-vs-all” 2x2 tables

Predicted




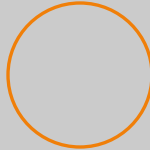

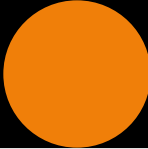
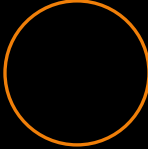

Actual



| Look for ● | Actually Target | Actually Not Target | Look for ○ | Actually Target | Actually Not Target |
|------------------------------|--------------------|------------------------|------------------------------|--------------------|------------------------|
| Selected/G uessed | 2 | 1 | Selected/G uessed | 2 | 1 |
| Not select/not guessed | 2 | 4 | Not select/not guessed | 1 | 5 |

| Look for ◻ | Actually Target | Actually Not Target |
|------------------------------|--------------------|------------------------|
| Selected/G uessed | 1 | 2 |
| Not select/not guessed | 1 | 5 |

2. Generalizing the 2-by-2 contingency table

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | # | # | # |
| |  | # | # | # |
| |  | # | # | # |

This is also called a **Confusion Matrix**


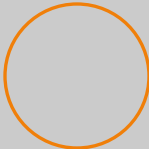


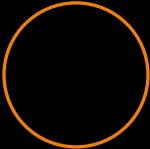

2. Generalizing the 2-by-2 contingency table

Predicted



Actual



| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | a # | b # | c # |
| |  | d # | e # | f # |
| |  | g # | h # | i # |





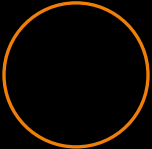

2. Generalizing the 2-by-2 contingency table

Predicted



Actual



| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | a 2 | b 0 | c 1 |
| |  | d 1 | e 2 | f 0 |
| |  | g 1 | h 1 | i 1 |

2. Generalizing the 2-by-2 contingency table

Predicted





















Actual





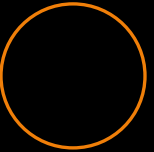



| | | Correct Value | | |
|---------------|--|---------------|--------|--------|
| | | | | |
| Guessed Value | | a 2 | b 0 | c 1 |
| | | d 1 | e 2 | f 0 |
| | | g 1 | h 1 | i 1 |

How do you compute TP_{\bullet} ?

2. Generalizing the 2-by-2 contingency table

| | | | | | | | | | |
|-----------|---|---|---|---|---|---|---|---|---|
| Predicted |  |  |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  |  |  |  |

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | 2 | 0 | 1 |
| |  | 1 | 2 | 0 |
| |  | 1 | 1 | 1 |

How do you compute TP_{\bullet} ?





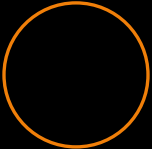

2. Generalizing the 2-by-2 contingency table

Predicted



Actual



| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | a 2 | b 0 | c 1 |
| |  | d 1 | e 2 | f 0 |
| |  | g 1 | h 1 | i 1 |

How do you compute FN ?

2. Generalizing the 2-by-2 contingency table

Predicted



Actual



| | | Correct Value | | |
|---------------|--|---------------|---|---|
| | | | | |
| Guessed Value | | 2 | 0 | 1 |
| | | 1 | 2 | 0 |
| | | 1 | 1 | 1 |

How do you compute FN ?


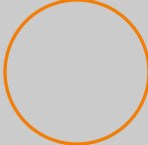


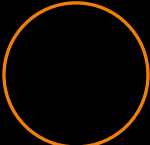

2. Generalizing the 2-by-2 contingency table

Predicted





















Actual





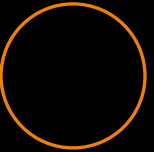



| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | a 2 | b 0 | c 1 |
| |  | d 1 | e 2 | f 0 |
| |  | g 1 | h 1 | i 1 |

How do you compute FP_{\circ} ?


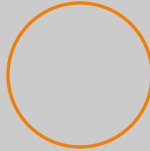


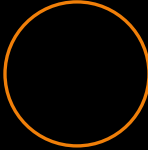

2. Generalizing the 2-by-2 contingency table

| | | | | | | | | | |
|-----------|---|---|---|---|---|---|---|---|---|
| Predicted |  |  |  |  |  |  |  |  |  |
| Actual |  |  |  |  |  |  |  |  |  |

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | 2 | 0 | 1 |
| |  | 1 | 2 | 0 |
| |  | 1 | 1 | 1 |


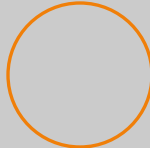


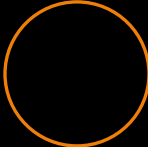

How do you compute FP_{\circ} ?

Performance of a Classifier using a Confusion Matrix

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | 80 | 9 | 11 |
| |  | 7 | 86 | 7 |
| |  | 2 | 8 | 9 |


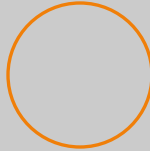


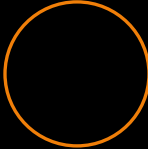

Q: Is this a good result?

Performance of a Classifier using a Confusion Matrix

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | 30 | 40 | 30 |
| |  | 25 | 30 | 50 |
| |  | 30 | 35 | 35 |

Q: Is this a good result?

Performance of a Classifier using a Confusion Matrix

| | | Correct Value | | |
|---------------|---|---|---|---|
| | |  |  |  |
| Guessed Value |  | 7 | 3 | 90 |
| |  | 4 | 8 | 88 |
| |  | 3 | 7 | 90 |

Q: Is this a good result?

Max Entropy / Logistic Regression Models

Outline

Maximum Entropy classifiers

Defining the model

Defining the objective

Learning: Optimizing the objective

Outline

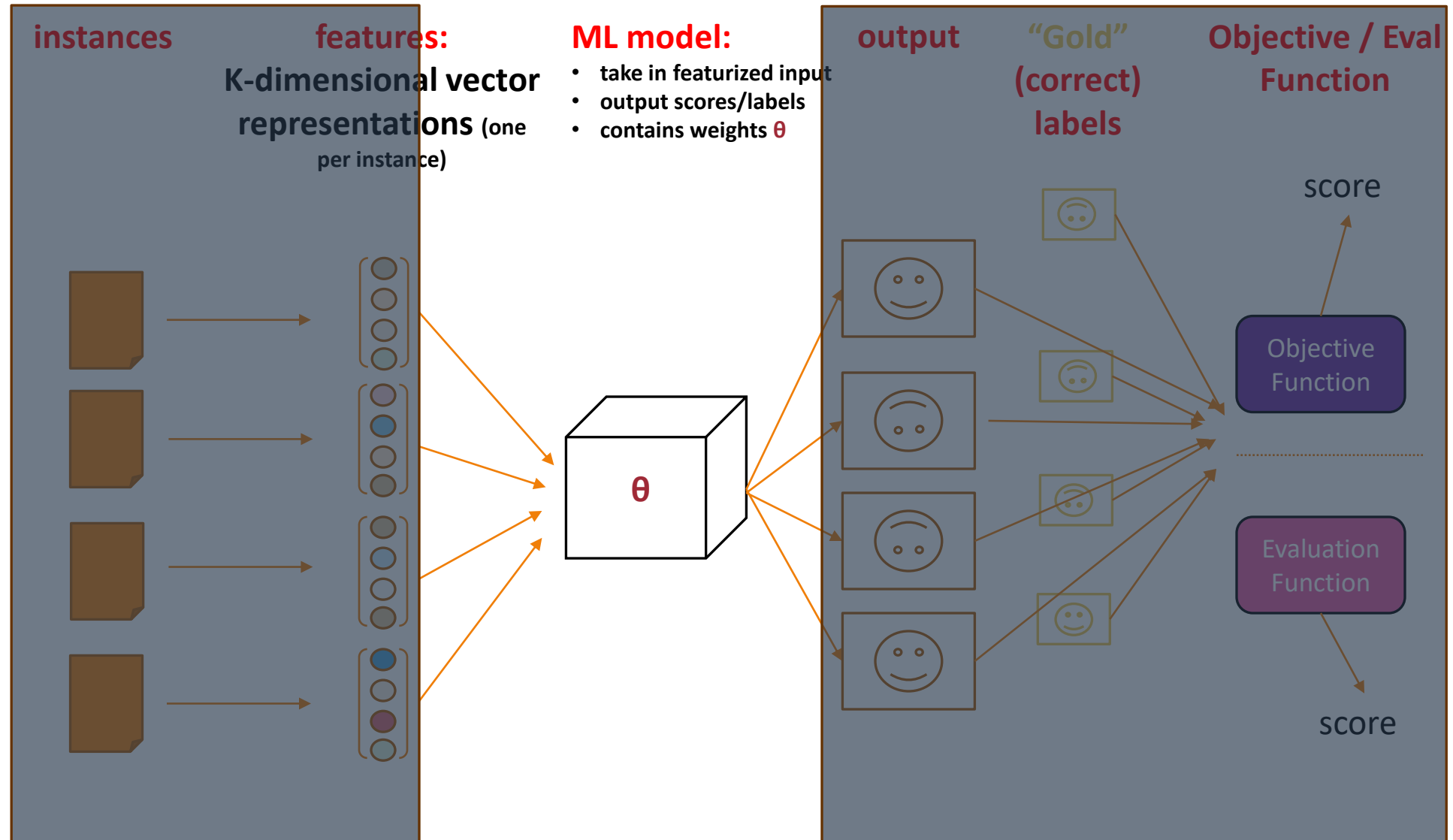
Maximum Entropy classifiers

Defining the model

Defining the objective

Learning: Optimizing the objective

Defining the Model



Terminology

| | |
|--------------------------------|---|
| common NLP term | Log-Linear Models |
| as statistical regression | (Multinomial) logistic regression Softmax regression |
| based in information theory | Maximum Entropy models (MaxEnt) |
| a form of | Generalized Linear Models |
| viewed as | Discriminative Naïve Bayes |
| to be cool today | Very shallow (sigmoidal) neural nets |

Maxent Models are Flexible

Maxent models can be used:

- to design discriminatively trained classifiers, or
- to create featureful language models

(among other approaches in NLP and ML more broadly)

Examining Assumption 3 Made for Classification Evaluation

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

$$p(\text{●} | X) \text{ vs. } p(\text{○} | X)$$

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



Key Take-away



We will *learn* this
 $p(Y | X)$

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

NLP pg. 477

Maxent Models for Classification: Discriminatively or ...

Directly model
the posterior

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

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)


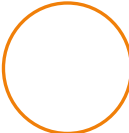
NLP pg. 478

Terminology: Posterior Probability

Posterior probability:

$$p(\text{●} | X) \text{ vs. } p(\text{○} | X)$$

Conditionally dependent probabilities:

- If  and  are the only two options:

$$p(\text{●} | X) + p(\text{○} | X) = 1$$

and

$$p(\text{●} | X) \geq 0, p(\text{○} | X) \geq 0$$

Posterior Probability with Variables

$$p(\text{●} | X) \text{ vs. } p(\text{○} | X)$$



$$p(Y = \text{label}_1 | X) \text{ vs. } p(Y = \text{label}_0 | X)$$

Maxent Models for Classification: Discriminatively or Generatively Trained

Directly model
the posterior

$$p(Y | X) = \text{maxent}(X; Y)$$

Discriminatively trained classifier


Model the
posterior with
Bayes rule

$$p(Y | X) \propto \text{maxent}(X | Y)p(Y)$$

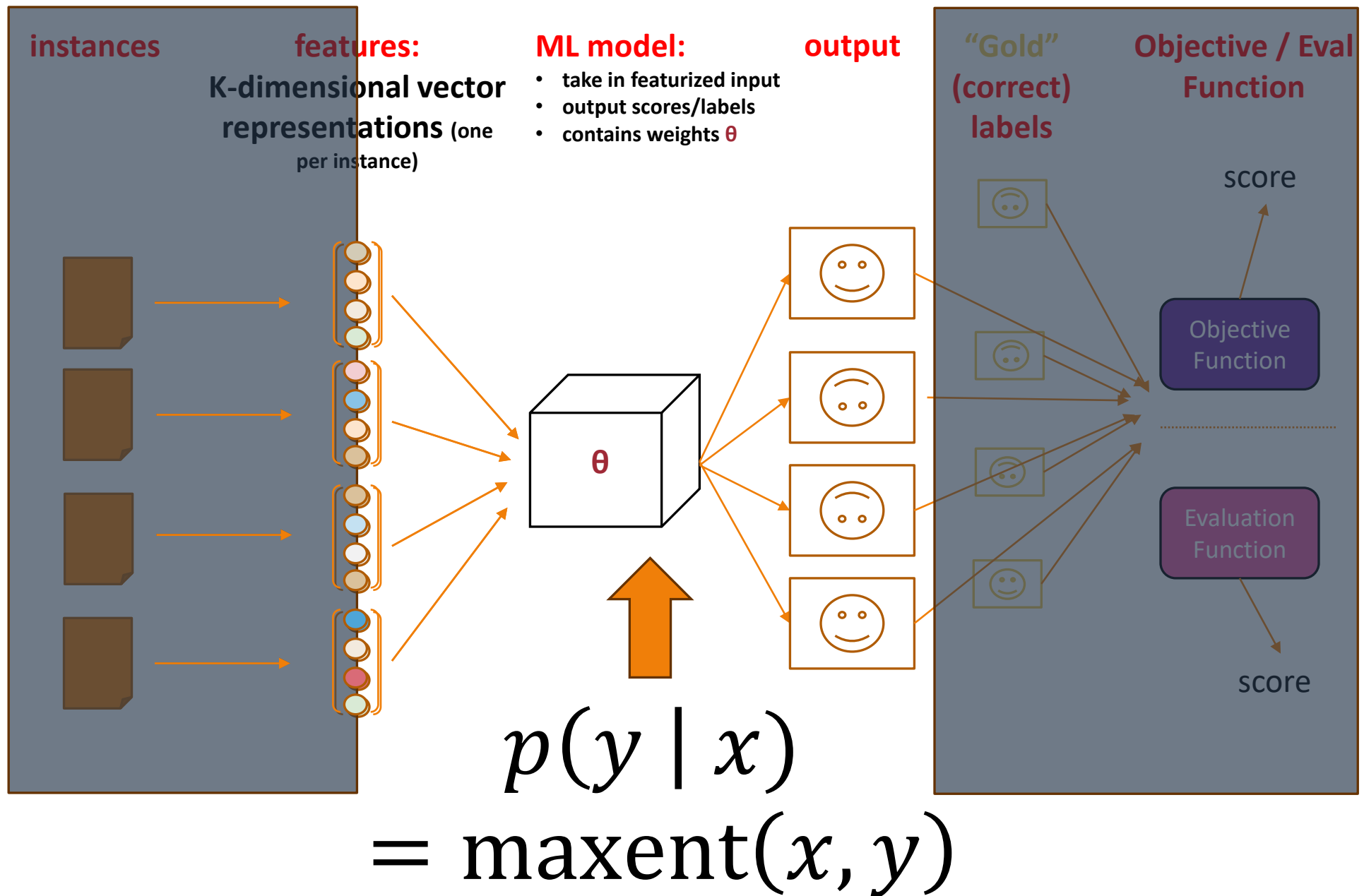
Generatively trained classifier with
maxent-based language model

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

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



Modeled
jointly!



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 $p(y \mid x)$

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

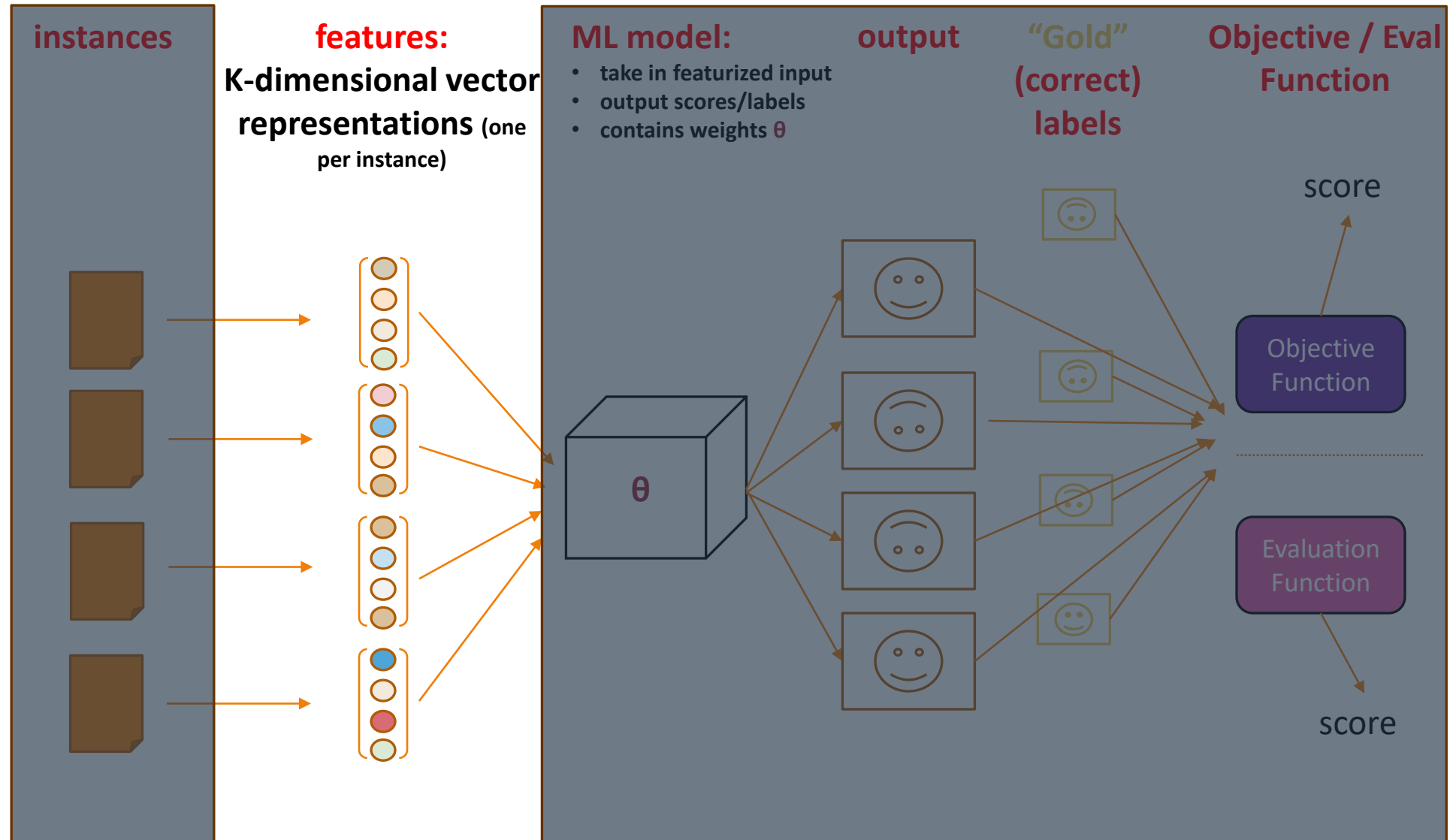
- Denoted by a **general vector of K features**

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

Features can be thought of as “soft” rules

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

Defining the Model



Review: Document Classification via Bag-of-Words Features (Example)

Amazon acquired MGM in 2022, taking over a sprawling library that includes more than 4,000 feature films and 17,000 television shows. The tech behemoth also earned the rights to distribute all the Bond movies, but the new deal solidifies the company's oversight of Bond's big-screen future.

With V word types, define V feature functions $f_i(x)$ as

$f_i(x)$ = # of times word type i appears in document x

$$f(x) = (f_i(x))_i^V$$

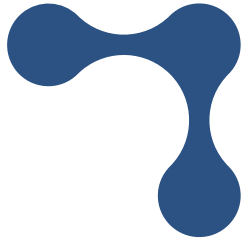
TECH
NOT TECH

Core assumption:
the label can be
predicted from
counts of individual
word types

| feature $f_i(x)$ | value |
|------------------|-------|
| Amazon | 1 |
| acquired | 1 |
| behemoth | 1 |
| Bond | 2 |
| ... | |
| sniffle | 0 |
| ... | |

$$f(x) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ \dots \end{bmatrix}$$

Example Task: Entailment



GLUE

<https://gluebenchmark.com/>

🤖 datasets: glue

| GLUE Tasks | |
|--|--------------------------|
| Name | Download |
| The Corpus of Linguistic Acceptability | Download |
| The Stanford Sentiment Treebank | Download |
| Microsoft Research Paraphrase Corpus | Download |
| Semantic Textual Similarity Benchmark | Download |
| Quora Question Pairs | Download |
| MultiNLI Matched | Download |
| MultiNLI Mismatched | Download |
| Question NLI | Download |
| Recognizing Textual Entailment | Download |
| Winograd NLI | Download |
| Diagnostics Main | Download |

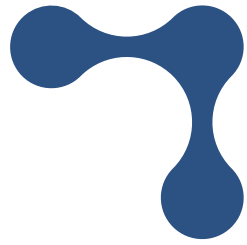
SuperGLUE 1

| Name | Identifier |
|--|------------|
| Broadcoverage Diagnostics | AX-b |
| CommitmentBank | CB |
| Choice of Plausible Alternatives | COPA |
| Multi-Sentence Reading Comprehension | MultiRC |
| Recognizing Textual Entailment | RTE |
| Words in Context | WiC |
| The Winograd Schema Challenge | WSC |
| BoolQ | BoolQ |
| Reading Comprehension with Commonsense Reasoning | ReCoRD |
| Winogender Schema Diagnostics | AX-g |

 **SuperGLUE**

<https://super.gluebenchmark.com/>

🤖 datasets: super_glue

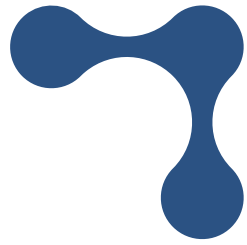


Recognizing Textual Entailment (RTE)

Given a premise sentence s and hypothesis sentence h ,
determine if h “follows from” s

ENTAILMENT (yes):

NOT ENTAILED (no):



Recognizing Textual Entailment (RTE)

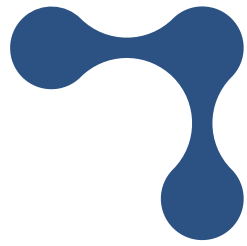
Given a premise sentence s and hypothesis sentence h ,
determine if h “follows from” s

ENTAILMENT (yes):

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 (no):



Recognizing Textual Entailment (RTE)

Given a premise sentence s and hypothesis sentence h ,
determine if h “follows from” s

ENTAILMENT (yes):

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 (no):

s : Based on a worldwide study of smoking-related fire and disaster data, UC Davis epidemiologists show smoking is a leading cause of fires and death from fires globally.

h : Domestic fires are the major cause of fire death.

RTE

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

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.

)

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.

ENTAILED

h: The Bulls basketball team is based in Chicago.

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**.

ENTAILED

These extractions are all **features** that have **fired** (likely have some significance)

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**.

ENTAILED

These extractions are all **features** that have **fired** (likely have some significance)

Discriminative Document Classification

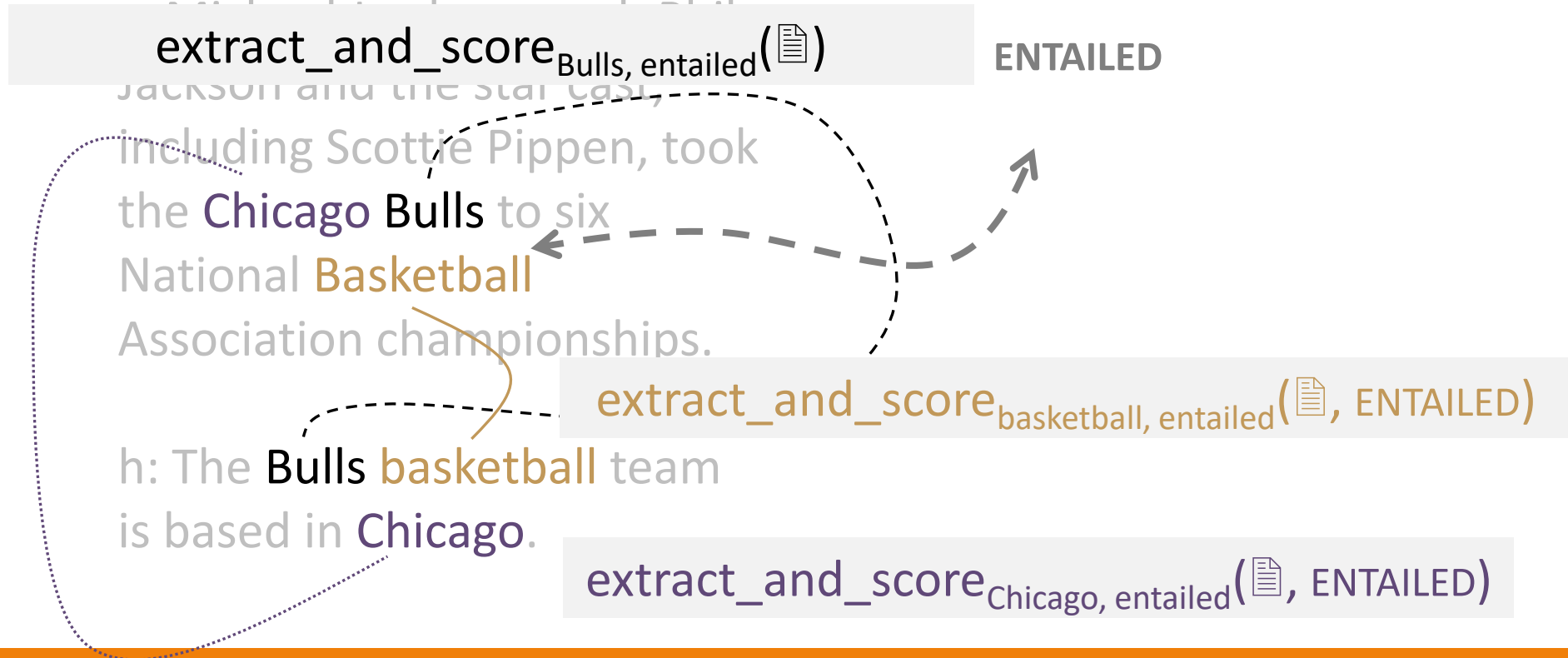
s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the **Chicago Bulls** to six National **Basketball** Association championships.

h: The **Bulls basketball** team is based in **Chicago**.

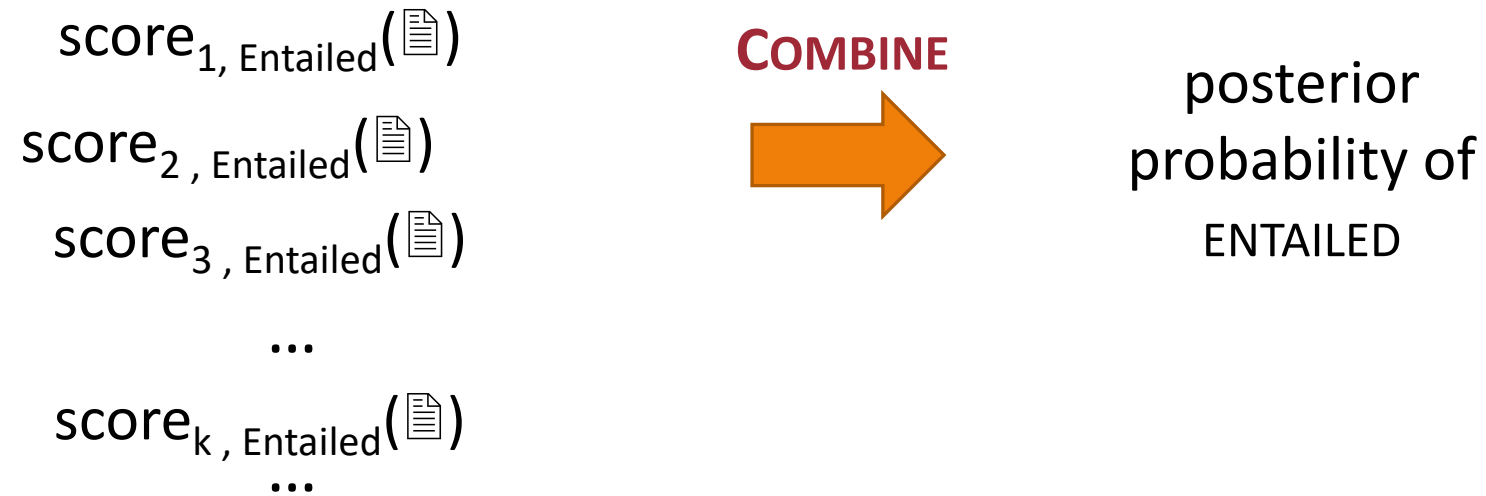
ENTAILED

These extractions are all **features** that have **fired** (likely have some significance)

We need to *score* the different extracted clues.



Score and Combine Our Clues



Scoring Our Clues

score(, 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.

*(ignore the
feature indexing
for now)*

score₁ , Entailed (📄)

+

score₂ , Entailed (📄)

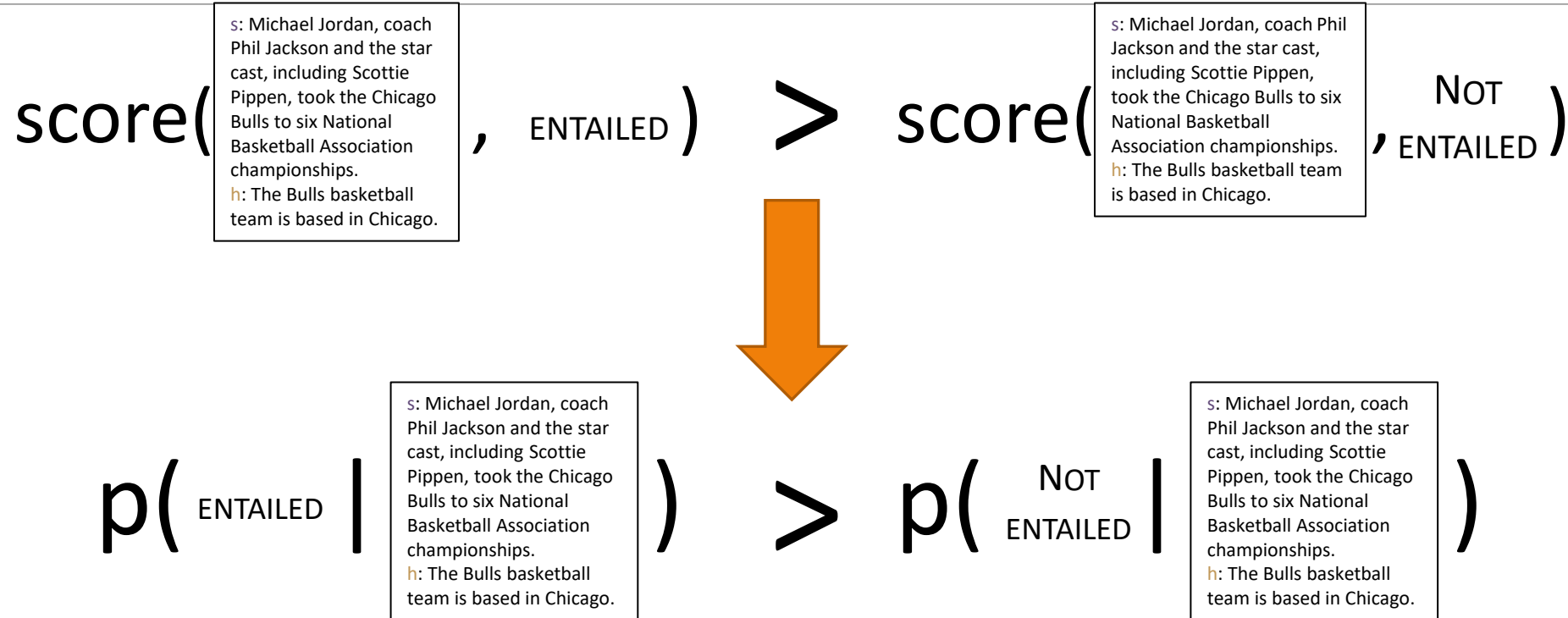
+

score₃ , Entailed (📄)

+

...

Turning Scores into Probabilities



KEY IDEA

Turning Scores into Probabilities (More Generally)

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



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

KEY IDEA

Maxent Modeling

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

Convert through
function G ?

What is this
function?

This must be a probability

This could be any real number

$G(\text{score}(\text{ENTAILED}))$

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.

What function G...

operates on any real number?

is never less than 0?

is monotonic? (if $a < b$, then $G(a) < G(b)$)

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{[Box]}) \propto \text{[Box]}$$

$$\exp(\text{score}(\text{[Box]}, \text{ENTAILED}))$$

Proportional to

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.

Maxent Modeling

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

$$\exp\left(\begin{array}{l} \text{score}_{1, \text{Entailed}}(\text{document icon}) + \\ \text{score}_{2, \text{Entailed}}(\text{document icon}) + \\ \text{score}_{3, \text{Entailed}}(\text{document icon}) + \\ \dots \end{array}\right)$$

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{...}) \propto$$

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.

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

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{...}) \propto$$

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.

$$\exp\left(\begin{aligned} &\text{weight}_{1, \text{Entailed}} * \text{applies}_1(\text{...}) + \\ &\text{weight}_{2, \text{Entailed}} * \text{applies}_2(\text{...}) + \\ &\text{weight}_{3, \text{Entailed}} * \text{applies}_3(\text{...}) + \\ &\dots \end{aligned}\right)$$

K different
weights...

for K different
features

$$\begin{bmatrix} \theta \\ .31 \\ -.5 \\ .1 \\ .002 \\ .522 \\ \dots \end{bmatrix}$$

$$\begin{bmatrix} f(x) \\ 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ \dots \end{bmatrix}$$

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{...}) \propto$$

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.

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

K different
weights...

for K different
features

multiplied and then summed

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{...}) \propto$$

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.

$$\exp(\text{Dot_product of Entailed weight_vec feature_vec}(\text{📄}))$$

K different
weights...

for K different
features

multiplied and
then summed

Maxent Modeling

$$p(\text{ENTAILED} \mid \text{...}) \propto$$

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

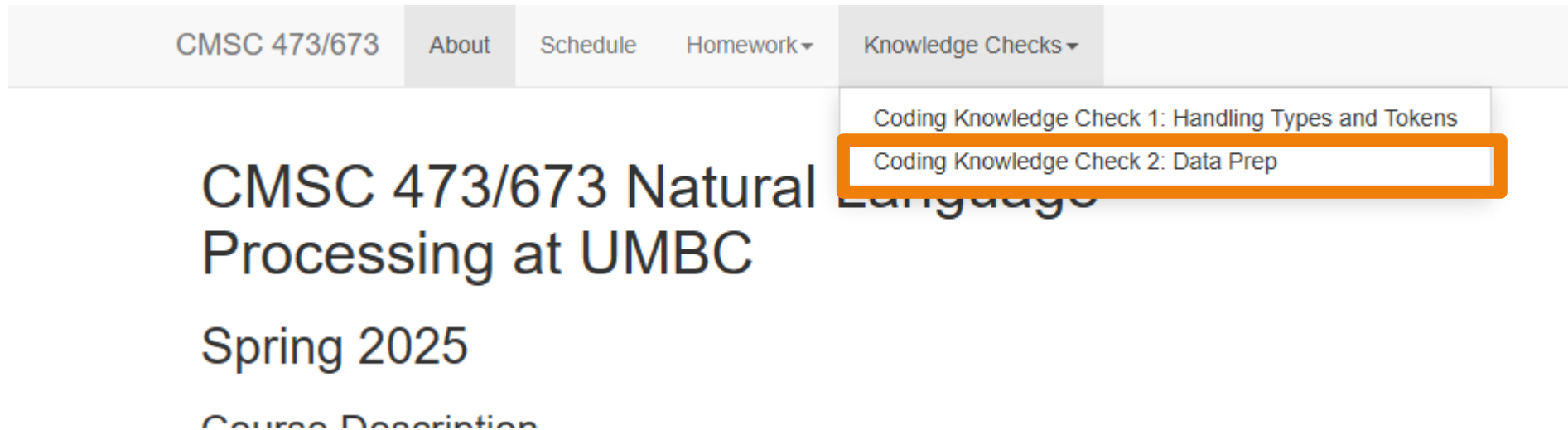
h: The Bulls basketball team is based in Chicago.

$$\exp\left(\theta_{\text{ENTAILED}}^T f(\text{document})\right) \times \begin{bmatrix} .31 & -.5 & .1 & .002 & .522 & \dots \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ \dots \end{bmatrix}$$

K different weights... for K different features multiplied and then summed

Knowledge Check: Data Prep

<https://colab.research.google.com/drive/19yg0EUXQtHozBiSuO6cKOBhoSPzQHgug?usp=sharing>



The image shows a navigation bar for the course 'CMSC 473/673'. The bar contains links for 'About', 'Schedule', 'Homework', and 'Knowledge Checks'. The 'Knowledge Checks' link is expanded, showing two options: 'Coding Knowledge Check 1: Handling Types and Tokens' and 'Coding Knowledge Check 2: Data Prep'. The second option is highlighted with an orange border. Below the navigation bar, the course title 'CMSC 473/673 Natural Language Processing at UMBC' and the semester 'Spring 2025' are visible.

CMSC 473/673

About Schedule Homework Knowledge Checks

Coding Knowledge Check 1: Handling Types and Tokens

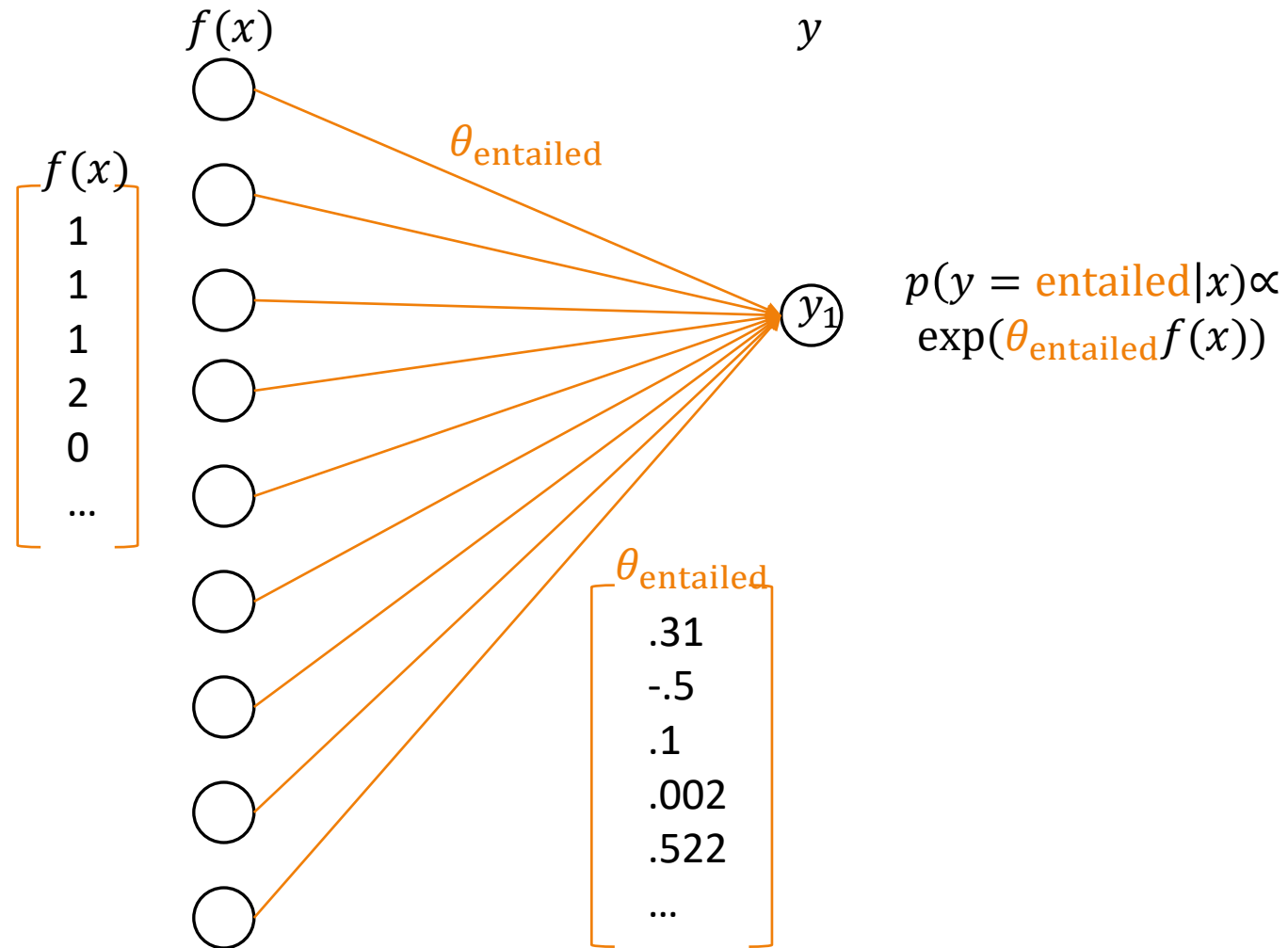
Coding Knowledge Check 2: Data Prep

CMSC 473/673 Natural Language Processing at UMBC

Spring 2025

Course Description

Maxent Classifier, schematically



Maxent Modeling

$$p(\text{ENTAILED} \mid \text{document}) \propto$$

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.

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

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.

$) =$

How do we define Z?

$\frac{1}{Z} \exp(\theta \text{ENTAILED} f(\text{document}))$

Normalization for Classification

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

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

classify doc x with label y in one go

Normalization for Classification (long form)

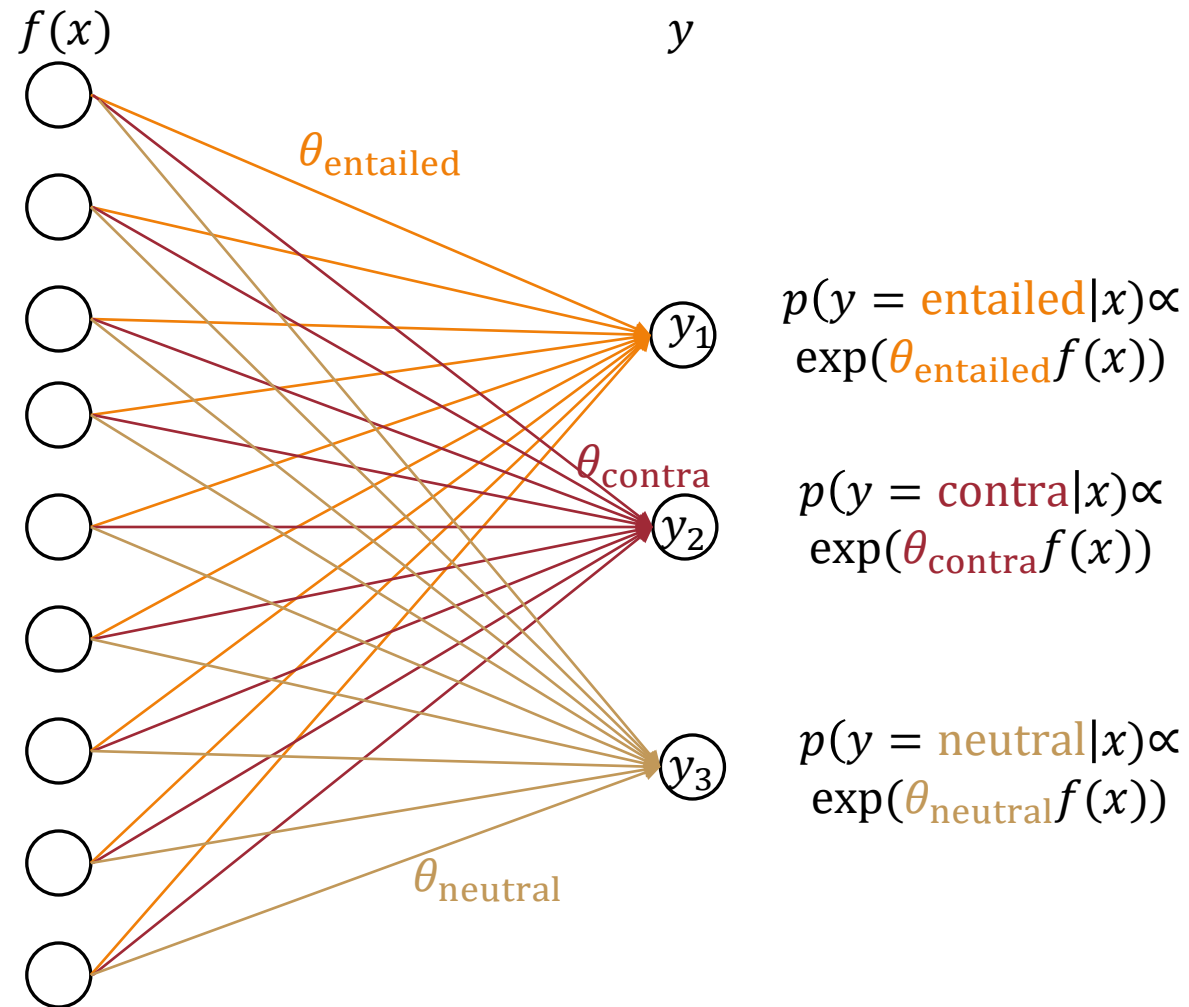
$$Z = \sum_{\text{label } j} \exp(\text{weight}_{1,j} * \text{applies}_1(\text{📄}) + \text{weight}_{2,j} * \text{applies}_2(\text{📄}) + \text{weight}_{3,j} * \text{applies}_3(\text{📄}) + \dots)$$

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

classify doc x with label y in one go

Maxent Classifier, schematically

Why would we want to normalize the weights?



output:
 $i = \text{argmax score}_i$
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

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

Different Notation, Same Meaning

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

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

$$p(Y | x) = \text{softmax}(\theta f(x))$$

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 a Linguistic “Blob”

User-defined

Integer representation/on e-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 | $n-1$ | $w_{i-n+1} \dots w_{i-1} w_i$ |

Templated Features

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

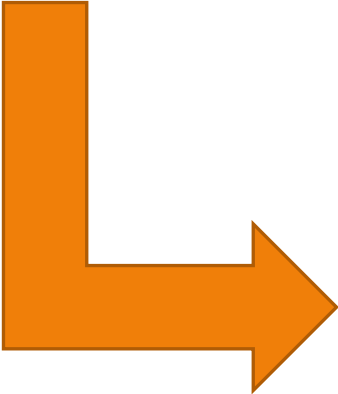
The feature `fclue` fires if the clue applies to/can be found in 

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

Maxent Modeling: Templated Binary Feature Functions

$$p(\text{ENTAILED} \mid \text{...}) \propto \exp\left(\text{weight}_{1, \text{Entailed}} * \text{applies}_1(\text{...}) + \text{weight}_{1, \text{Entailed}} * \text{applies}_2(\text{...}) + \text{weight}_{1, \text{Entailed}} * \text{applies}_3(\text{...}) + \dots\right)$$

$\text{applies}_{\text{target}}(\text{...}) = \begin{cases} 1, & \text{target matches } \text{...} \\ 0, & \text{otherwise} \end{cases}$
binary



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.

Example of a Templated Binary Feature Functions

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



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

Example of a Templated Binary Feature Functions

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



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

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

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



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

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

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



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

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

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



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

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

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



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

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

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



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

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

Defining the objective

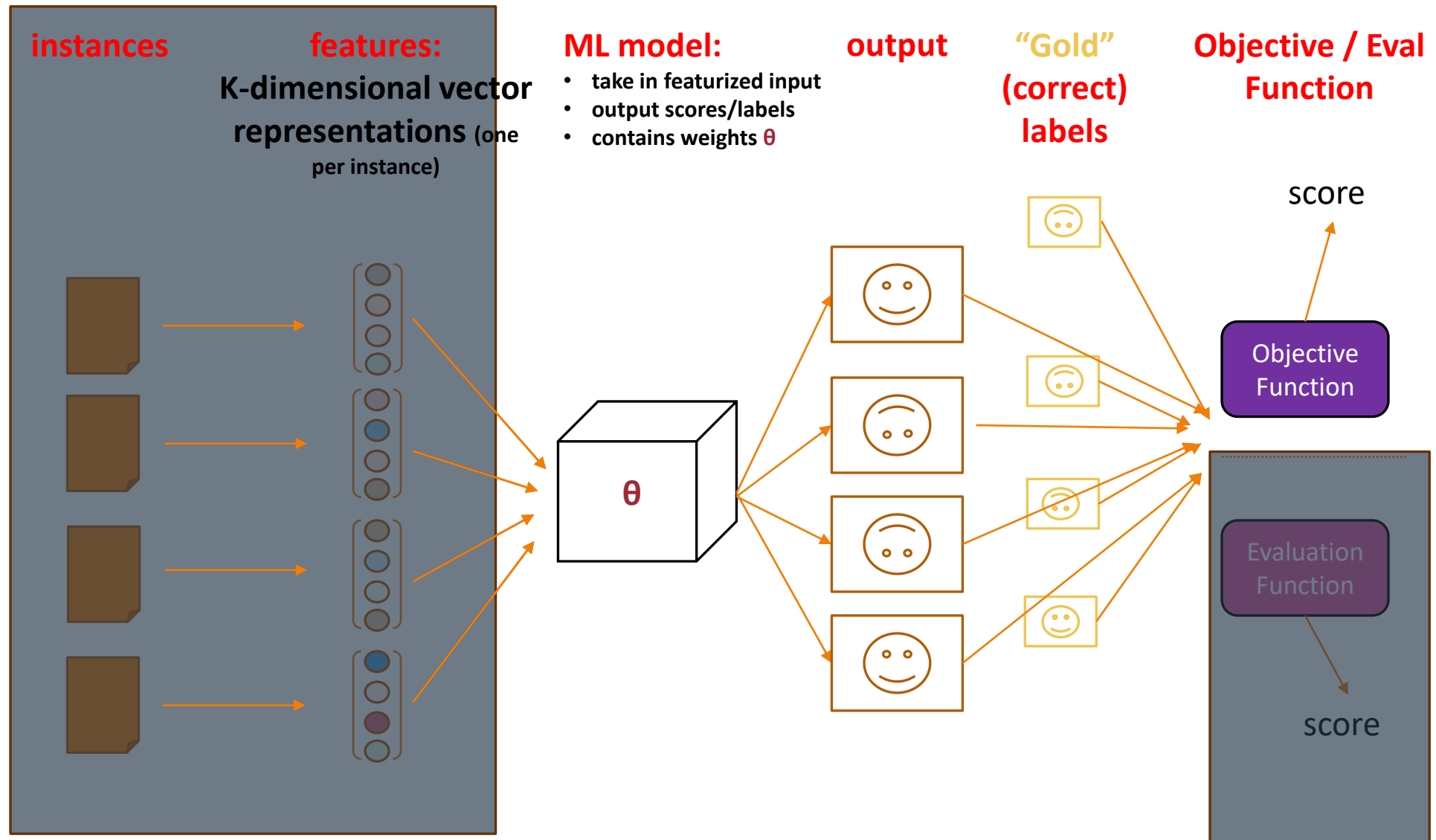
Learning: Optimizing the objective

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



$F(\theta; x, y)$ 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 $p(\text{label} \mid \text{document})$

For **language modeling**, this is $p(\text{word} \mid \text{history of words})$

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 \rightarrow 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)

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

Inverse of exp
 $\log(\exp(x)) = x$

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

Original maxent equation

$$\frac{\exp(\theta_y^T f(x))}{\sum_y \exp(\theta_y^T f(x))}$$

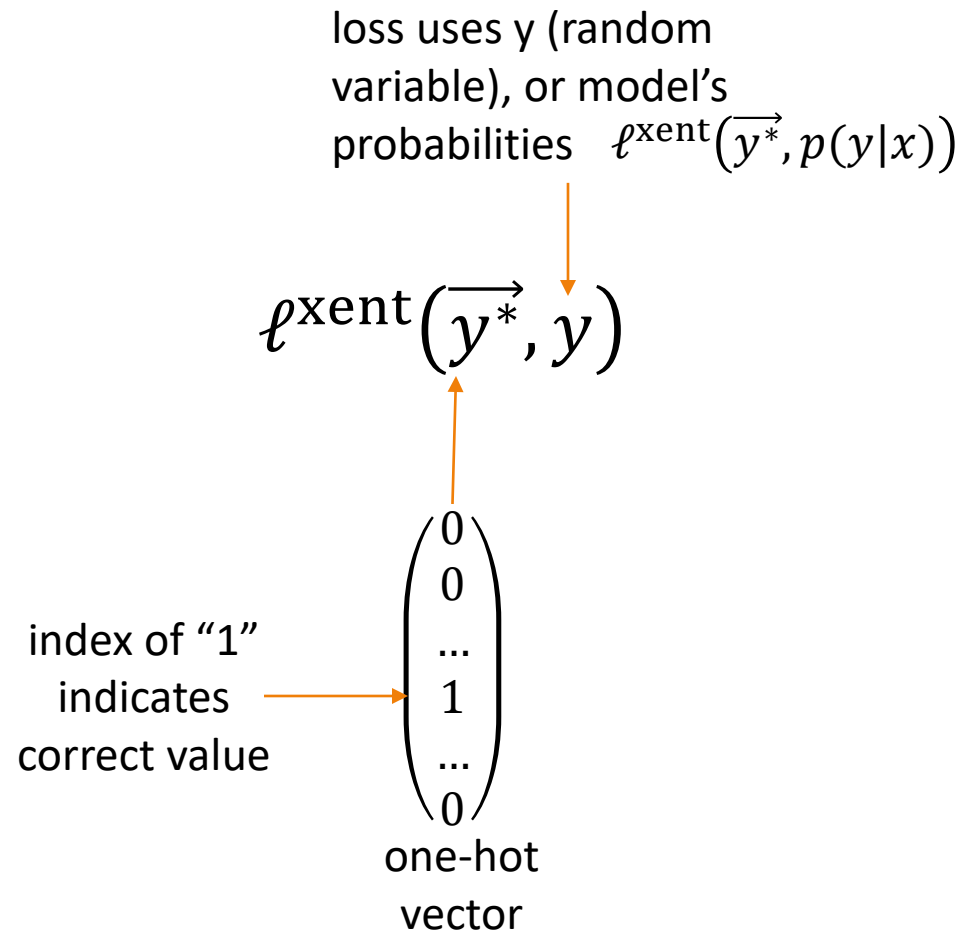
Differentiating this becomes nicer (even though Z depends on θ)

Log-Likelihood (Classification)

Wide range of (negative) numbers
Sums are more stable

$$\begin{aligned}\log \prod_i p_{\theta}(y_i | x_i) &= \sum_i \log p_{\theta}(y_i | x_i) \\ &= \sum_i \theta_{y_i}^T f(x_i) - \log Z(x_i) \\ &= F(\theta)\end{aligned}$$

Equivalent Version 2: *Minimize Cross Entropy Loss*



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 $\ell^{\text{xent}}(\vec{y}^*, p(y|x))$

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

index of "1"
indicates
correct value

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

one-hot
vector

Classification Log-likelihood \cong 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 $(minibatch, C, d_1, d_2, \dots, d_K)$ 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).

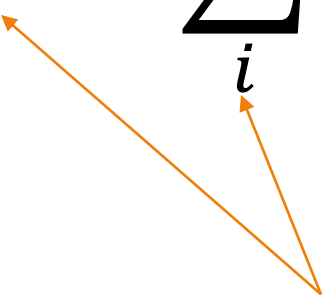
The loss can be described as:

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

Preventing Extreme Values

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

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



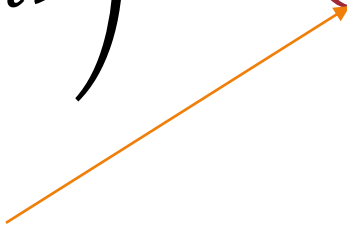
Learn the parameters based on
some (fixed) data/examples

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

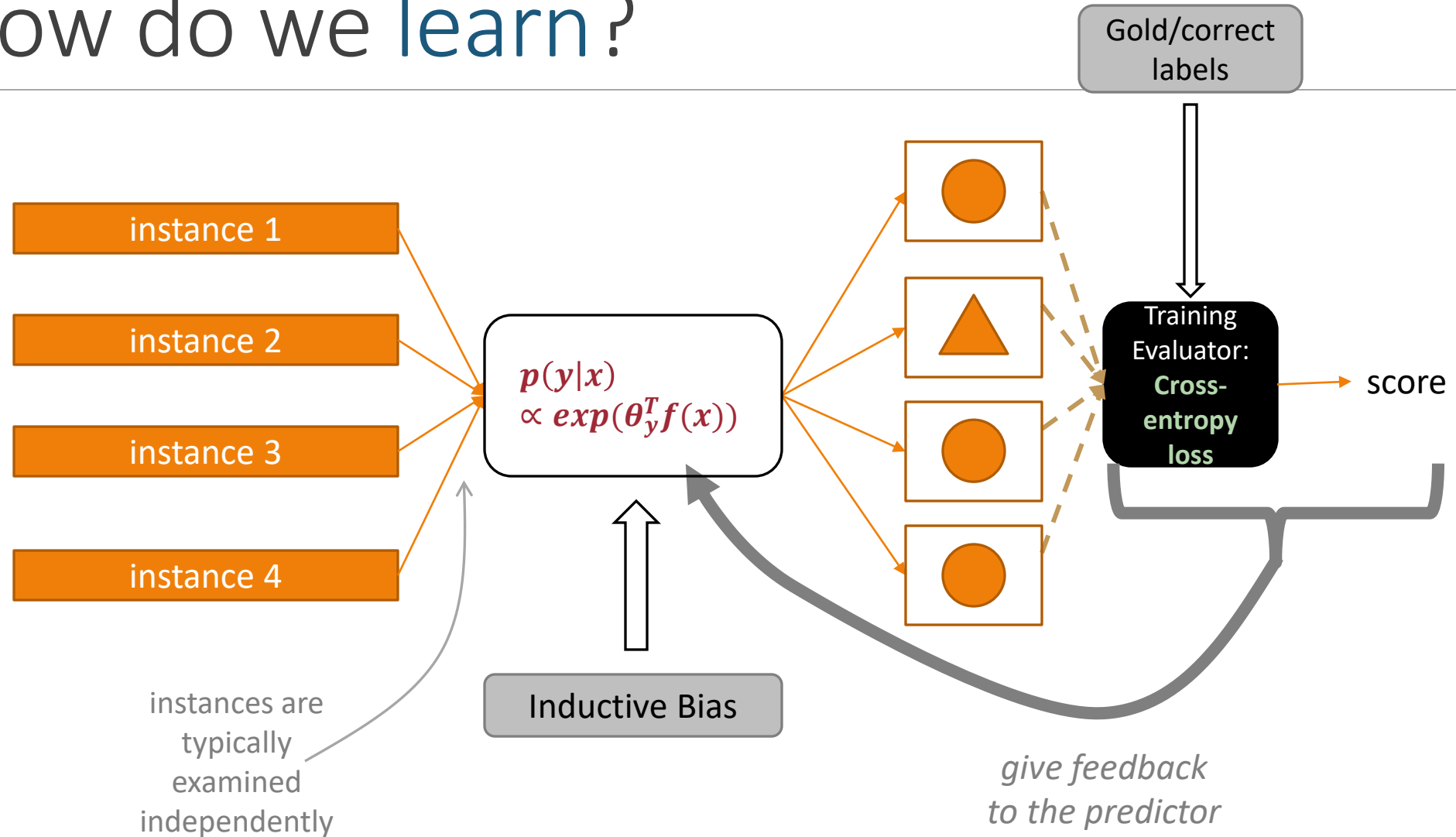
Maximum Entropy classifiers

Defining the model

Defining the objective

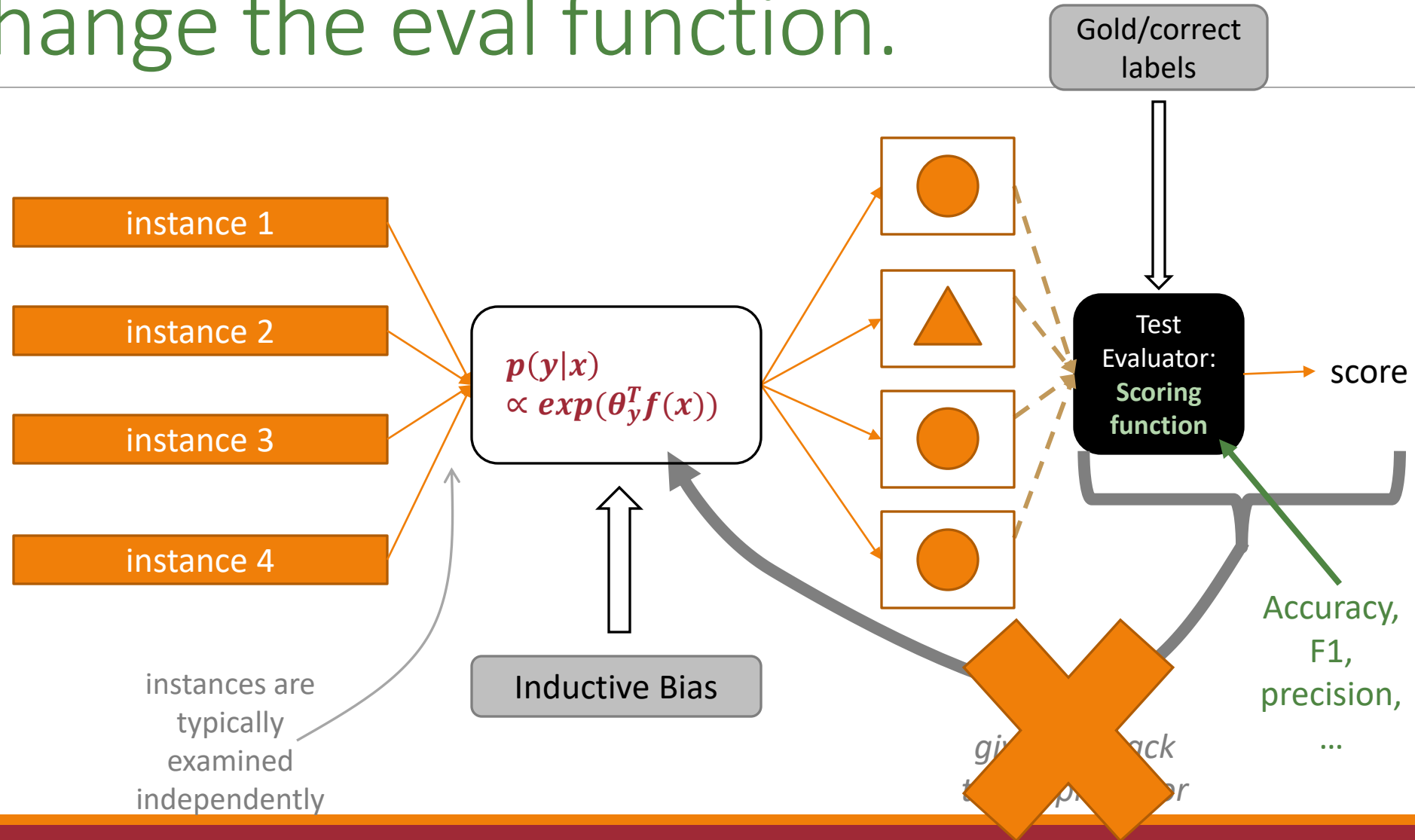
Learning: Optimizing the objective

How do we learn?



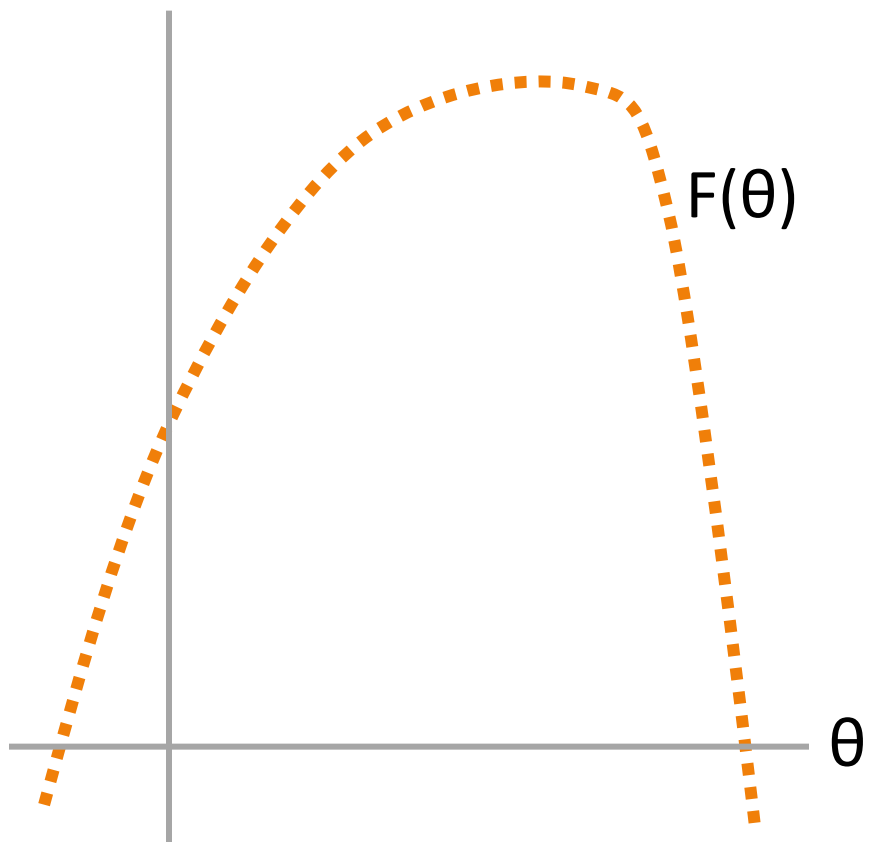
How do we evaluate (or use)?

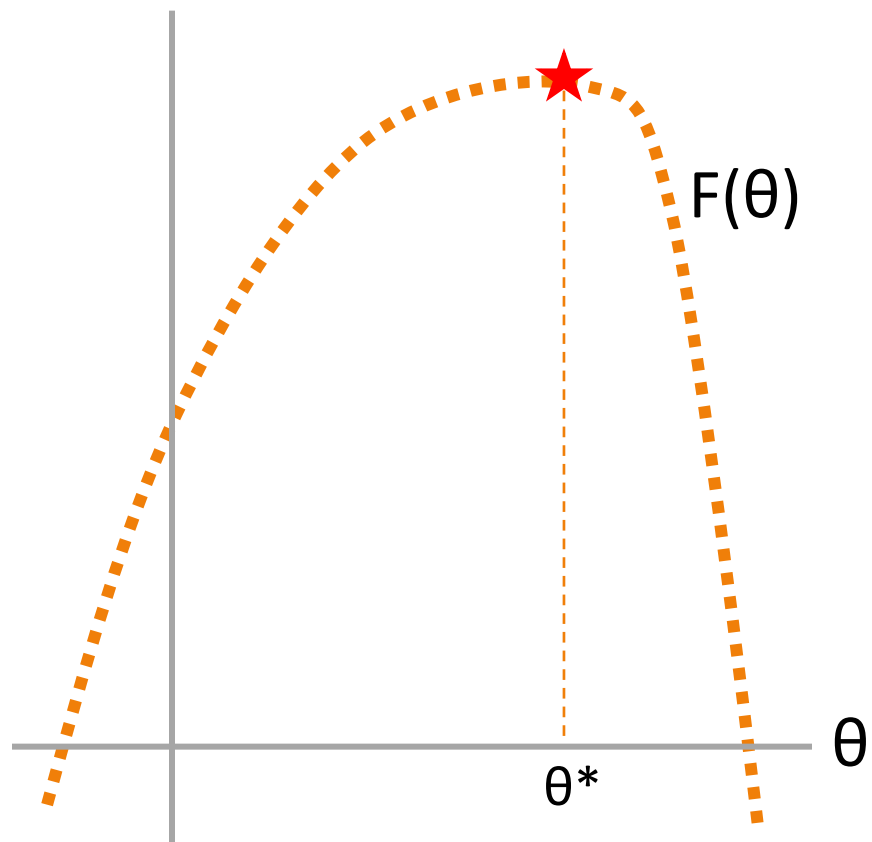
Change the eval function.

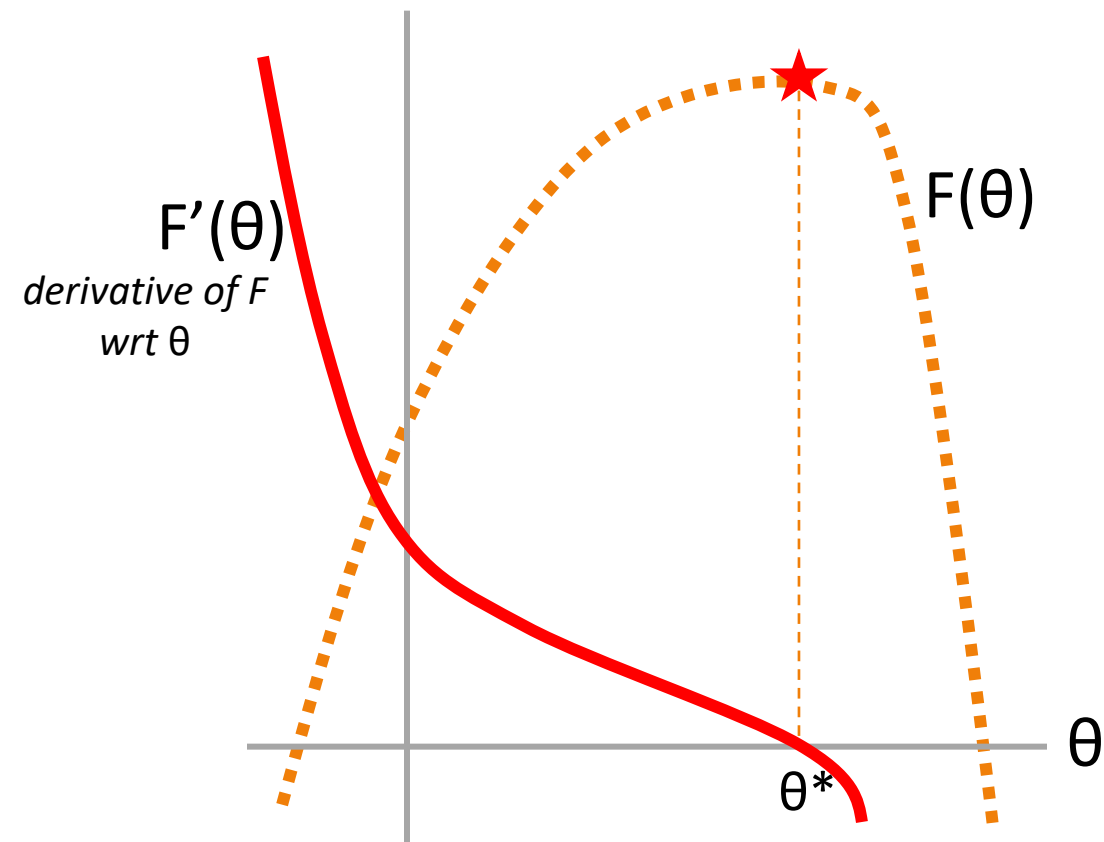


How will we optimize $F(\theta)$?

Calculus.







Example

(Best case, solve for roots of the derivative)

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



differentiate

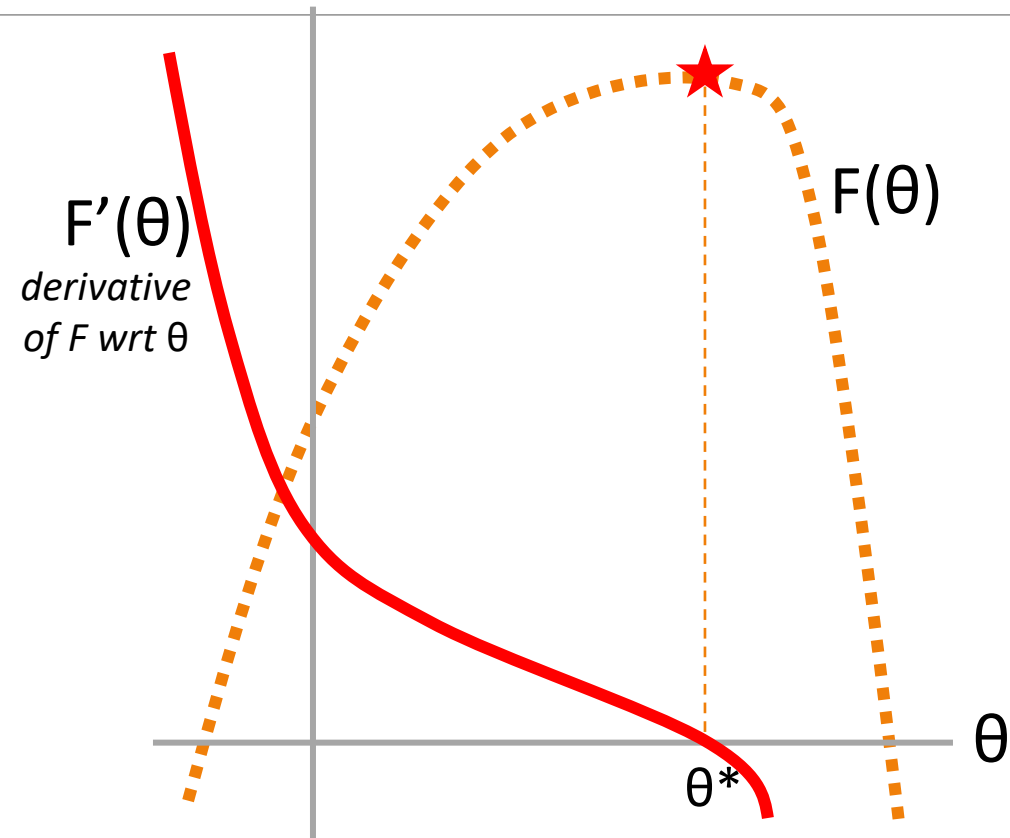
$$F'(x) = -2x + 4$$



Solve $F'(x) = 0$

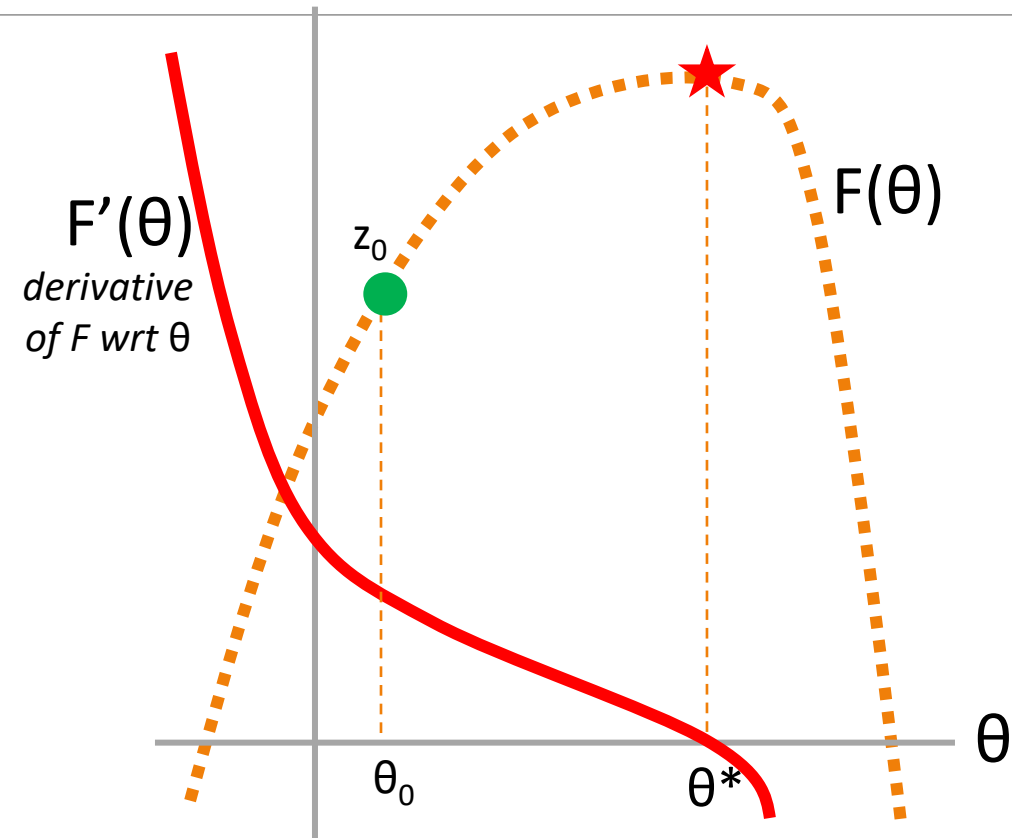
$$x = 2$$

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



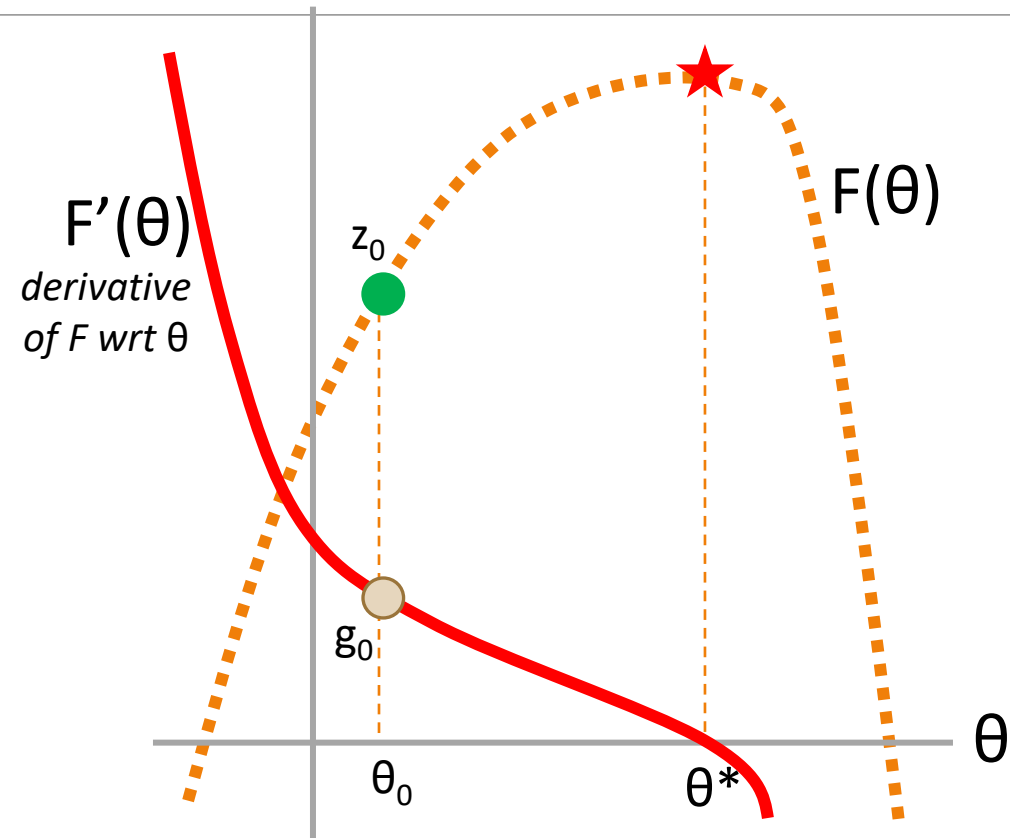
What if you can't find the roots? Follow the derivative

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 $g_t = F'(\theta_t)$



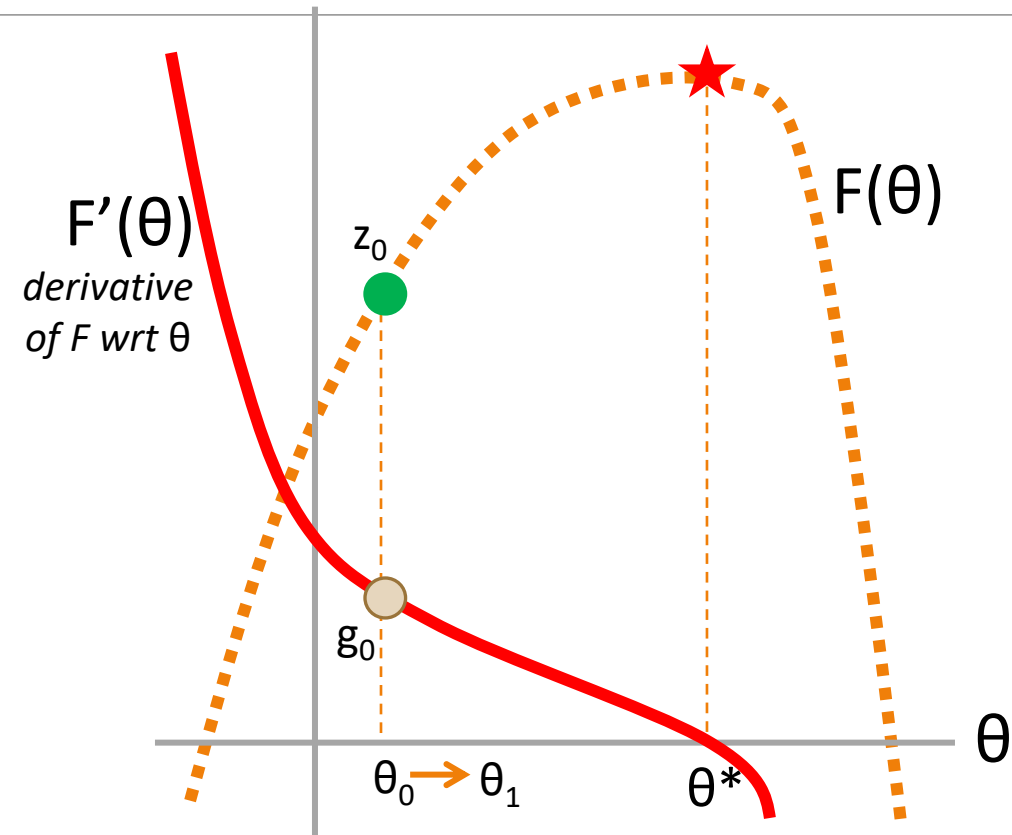
What if you can't find the roots? Follow the derivative

Set $t = 0$

Pick a starting value θ_t

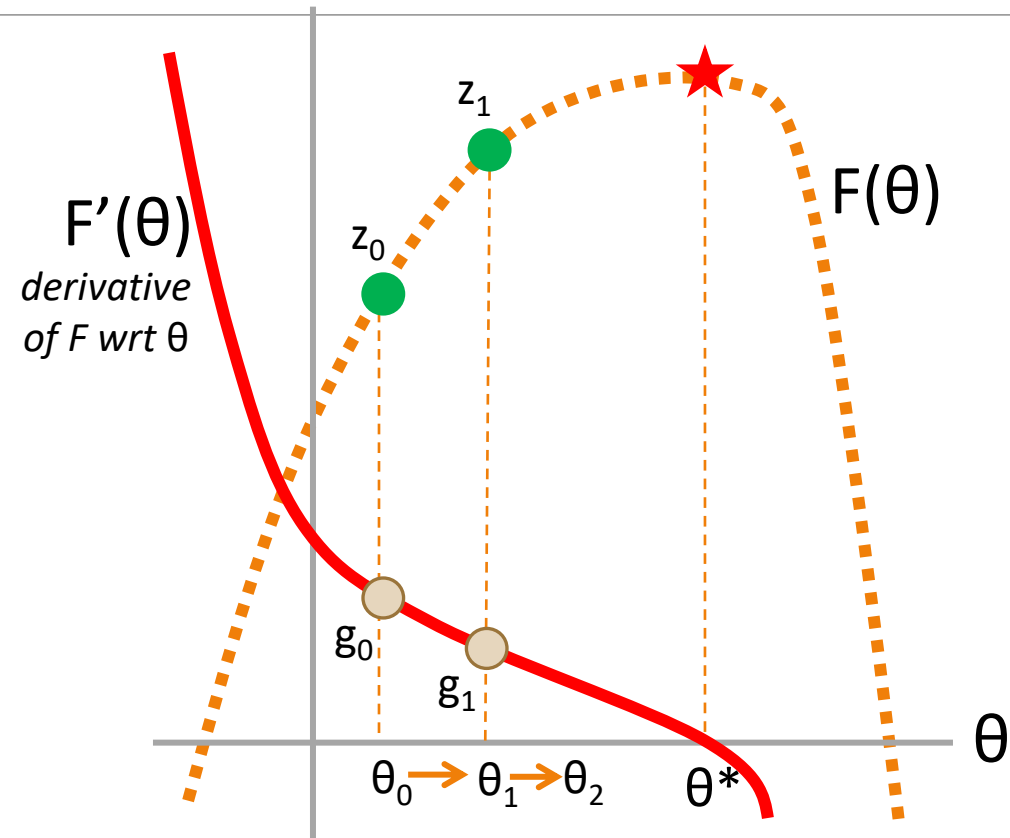
Until converged:

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 $t += 1$



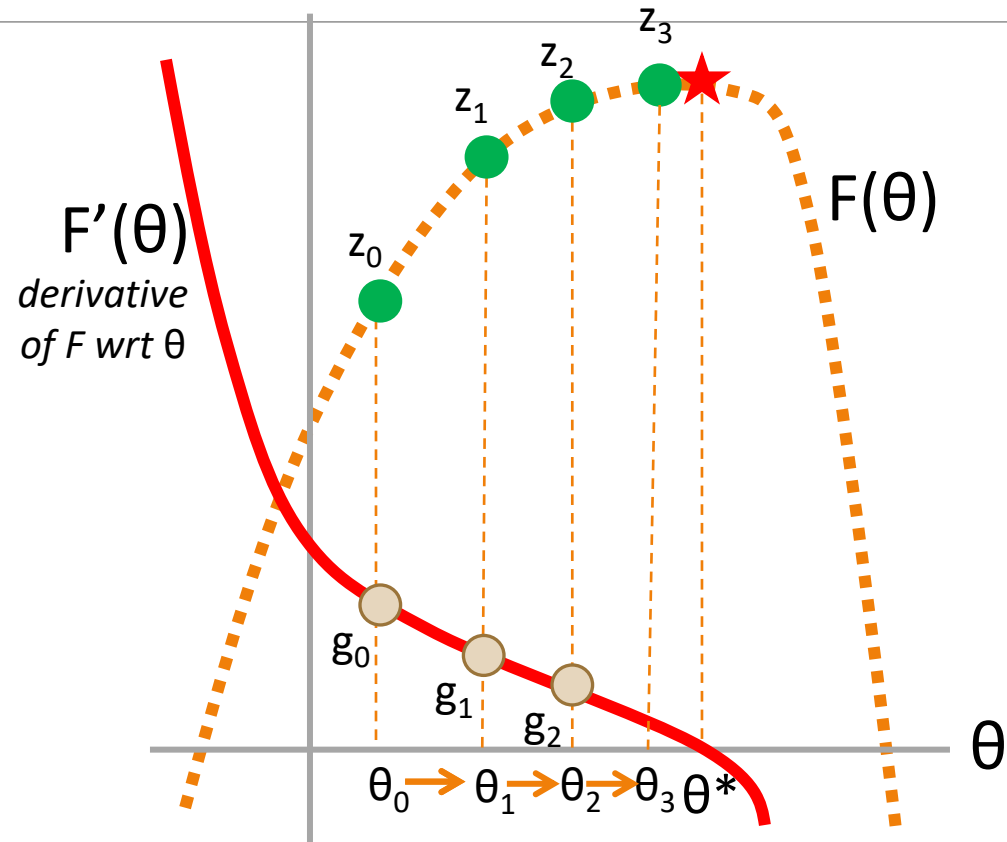
What if you can't find the roots? Follow the derivative

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 $t += 1$



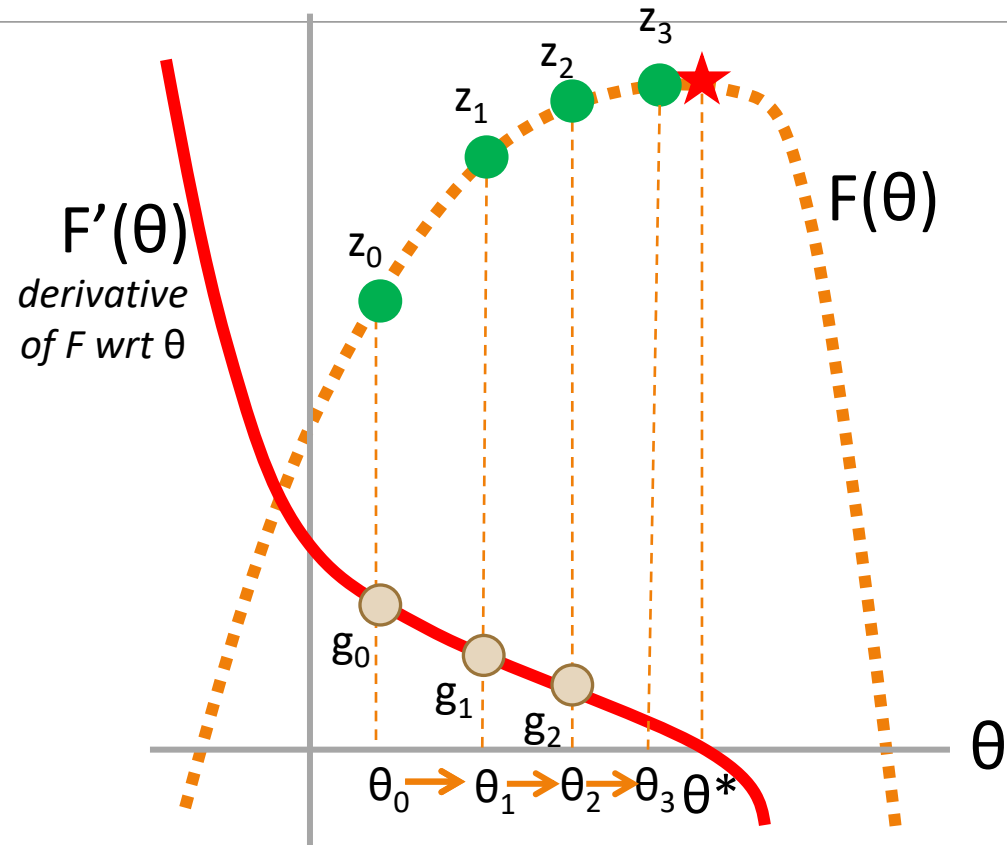
What if you can't find the roots? Follow the derivative

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 $t += 1$



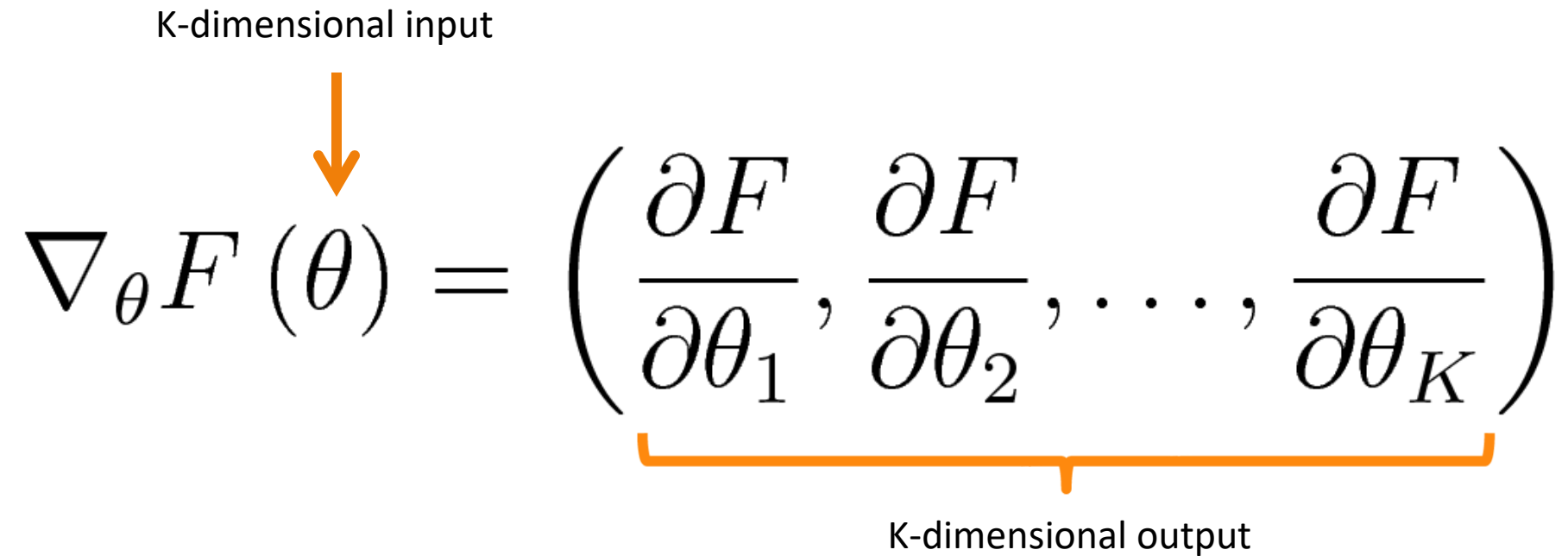
Gradient = Multi-variable derivative

K-dimensional input

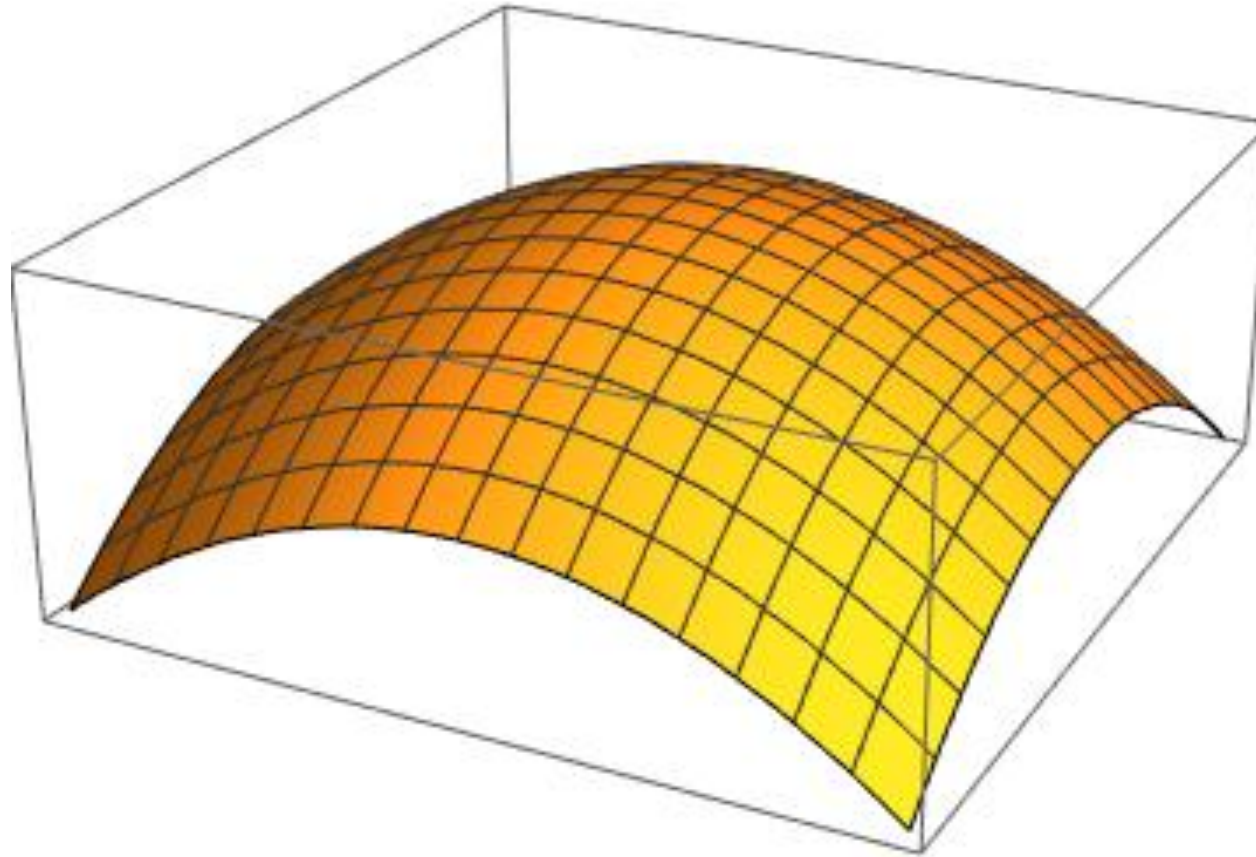
↓

$$\nabla_{\theta} F(\theta) = \left(\frac{\partial F}{\partial \theta_1}, \frac{\partial F}{\partial \theta_2}, \dots, \frac{\partial F}{\partial \theta_K} \right)$$

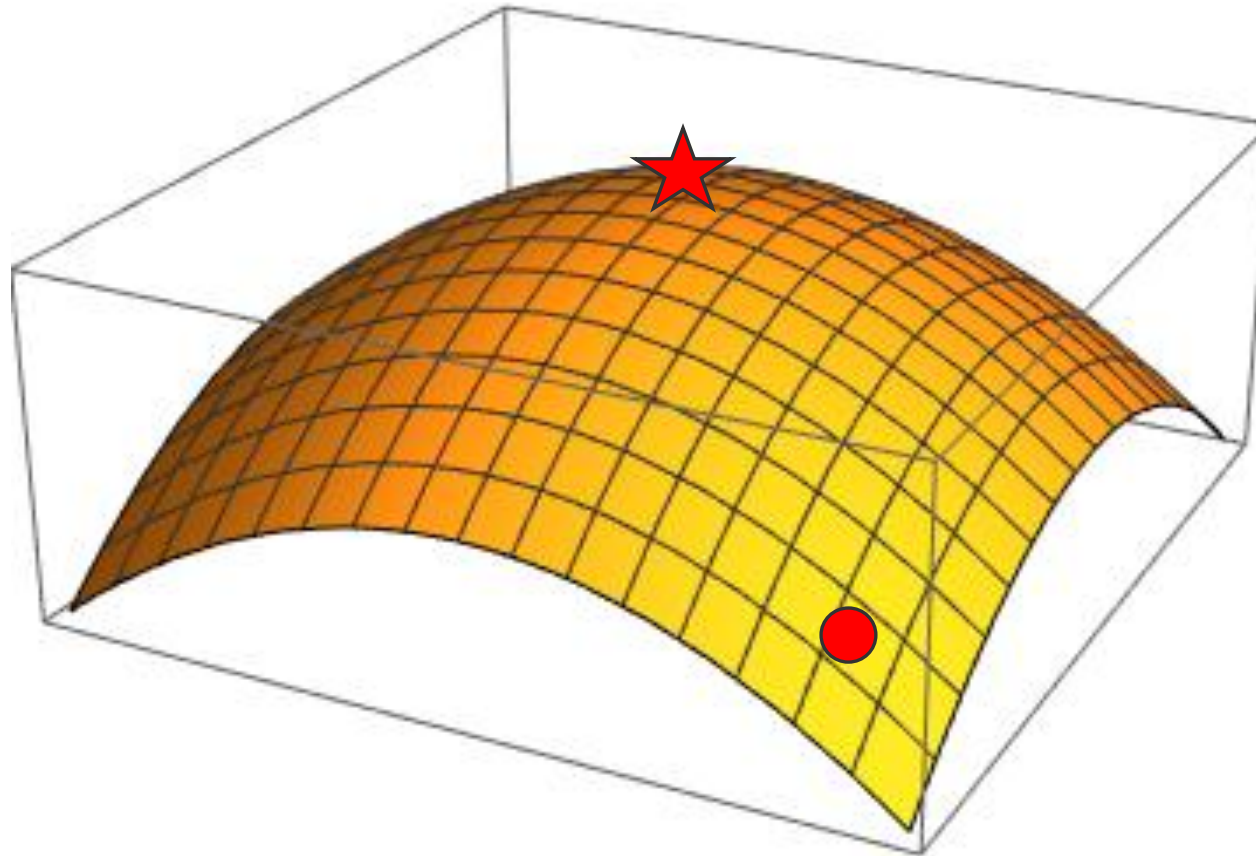
K-dimensional output



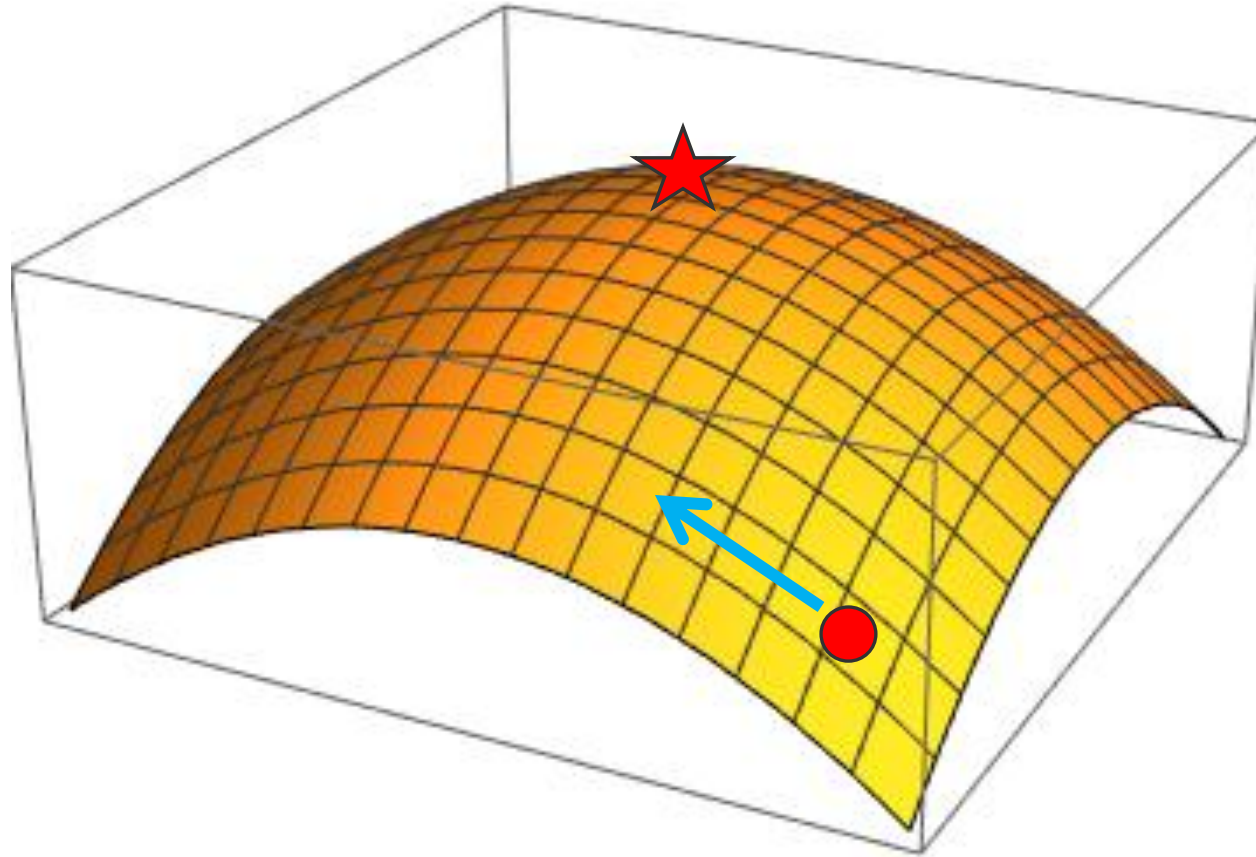
Gradient Ascent



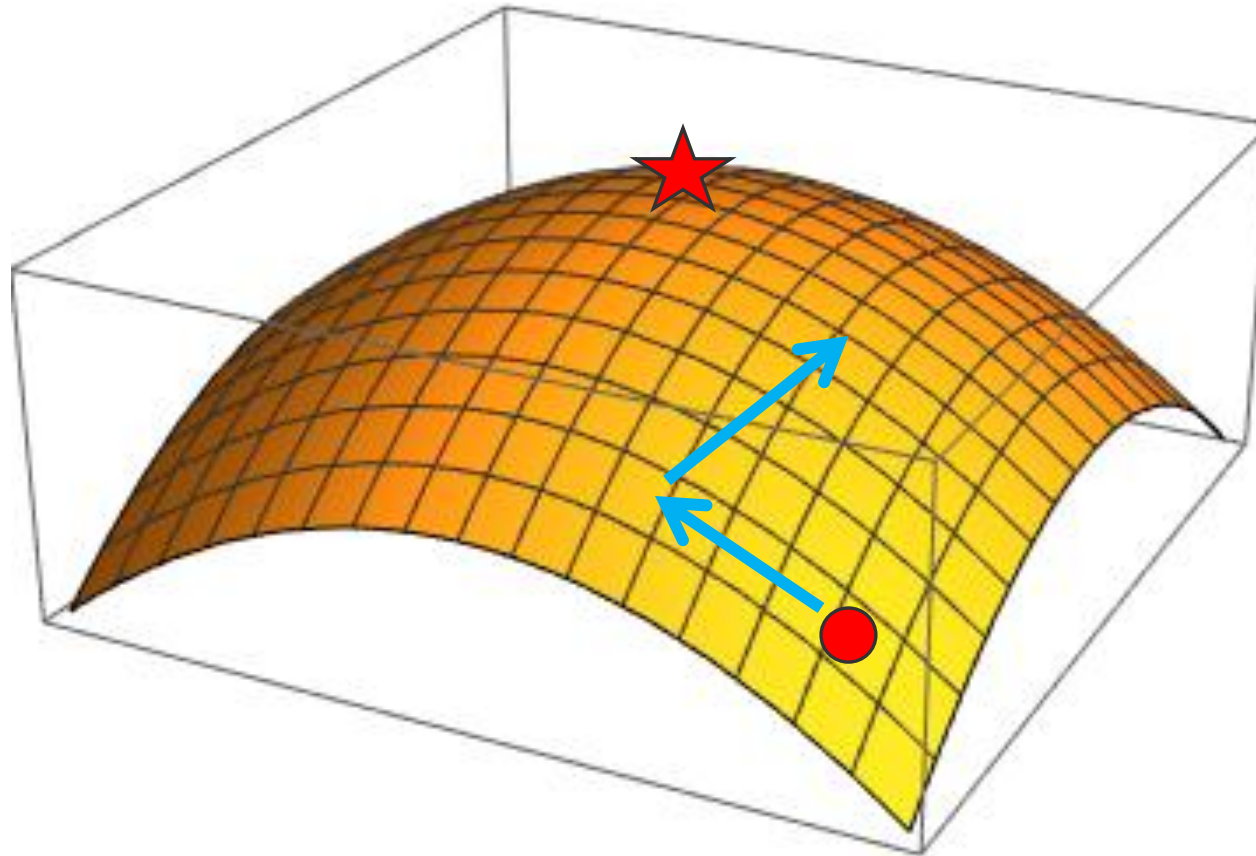
Gradient Ascent



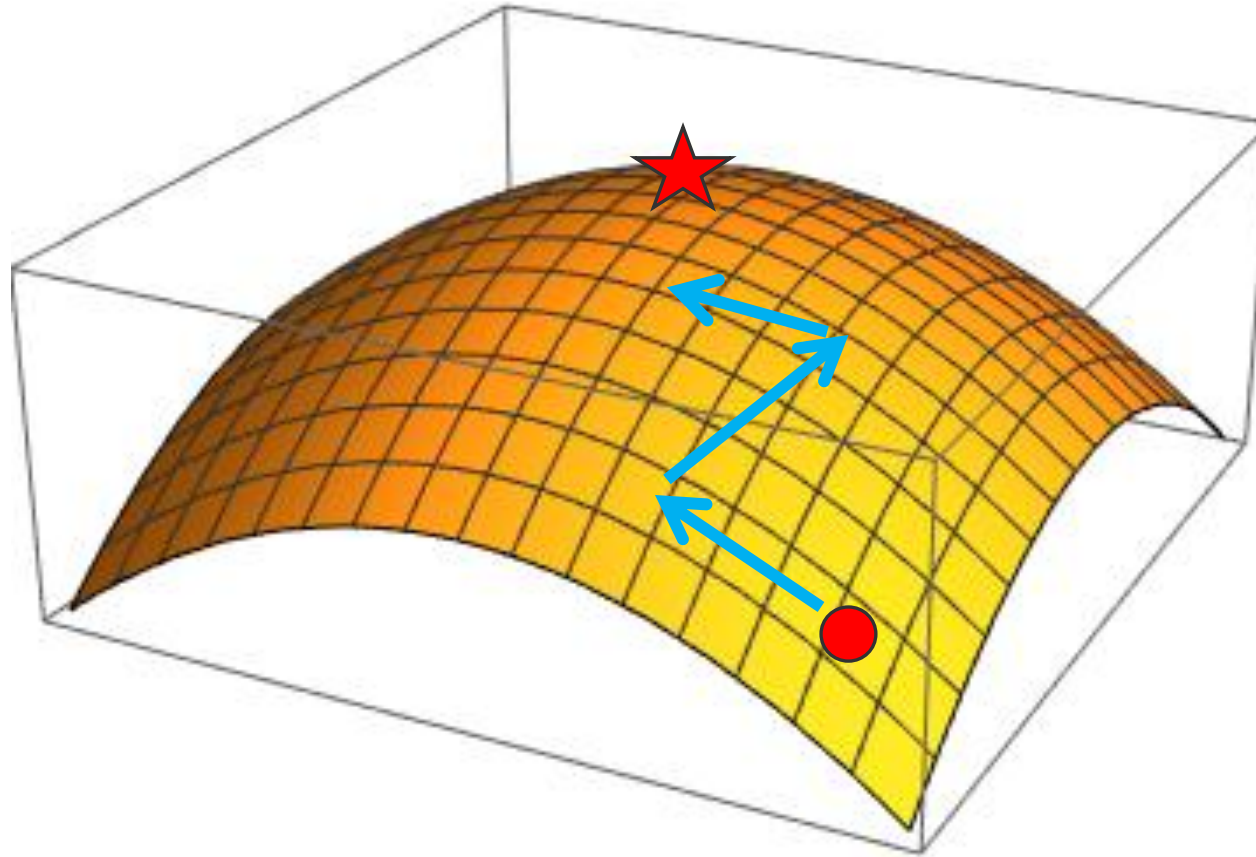
Gradient Ascent



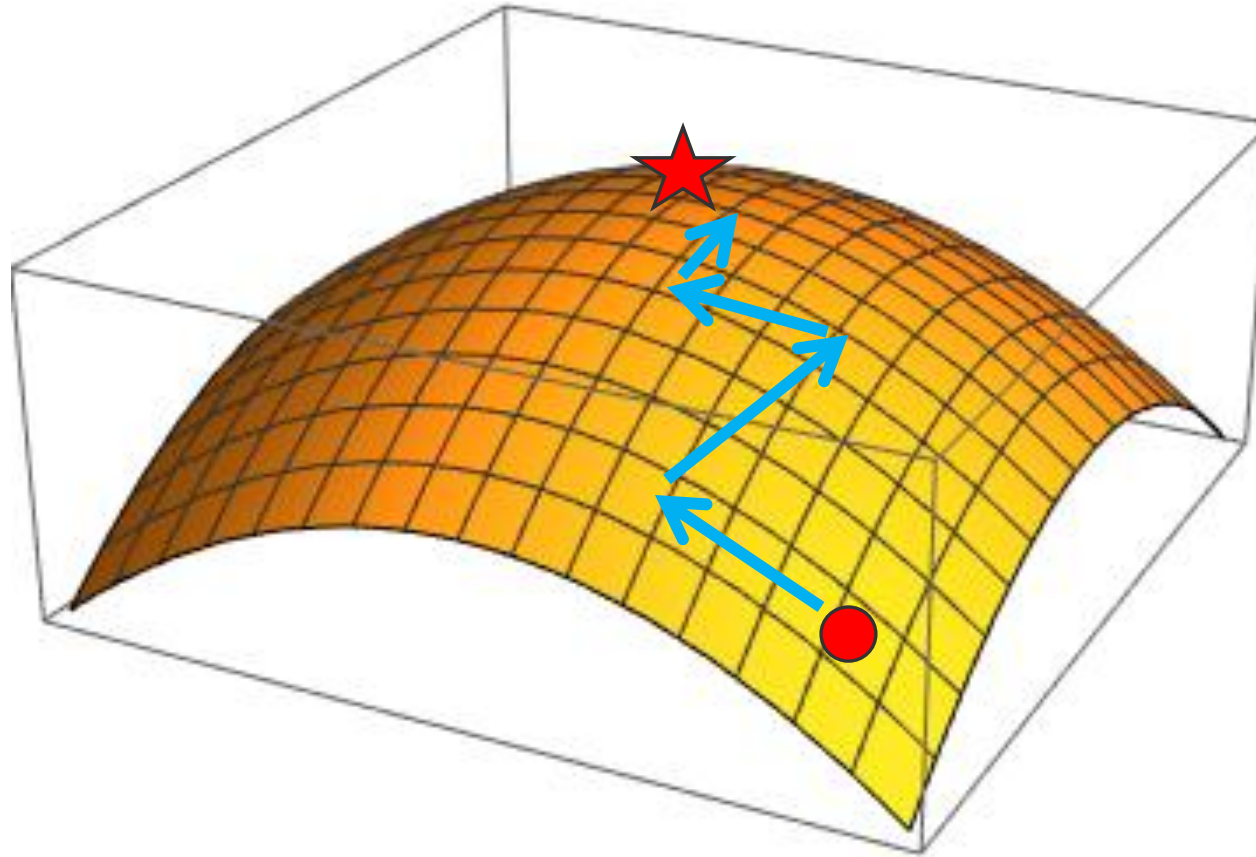
Gradient Ascent



Gradient Ascent



Gradient Ascent



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

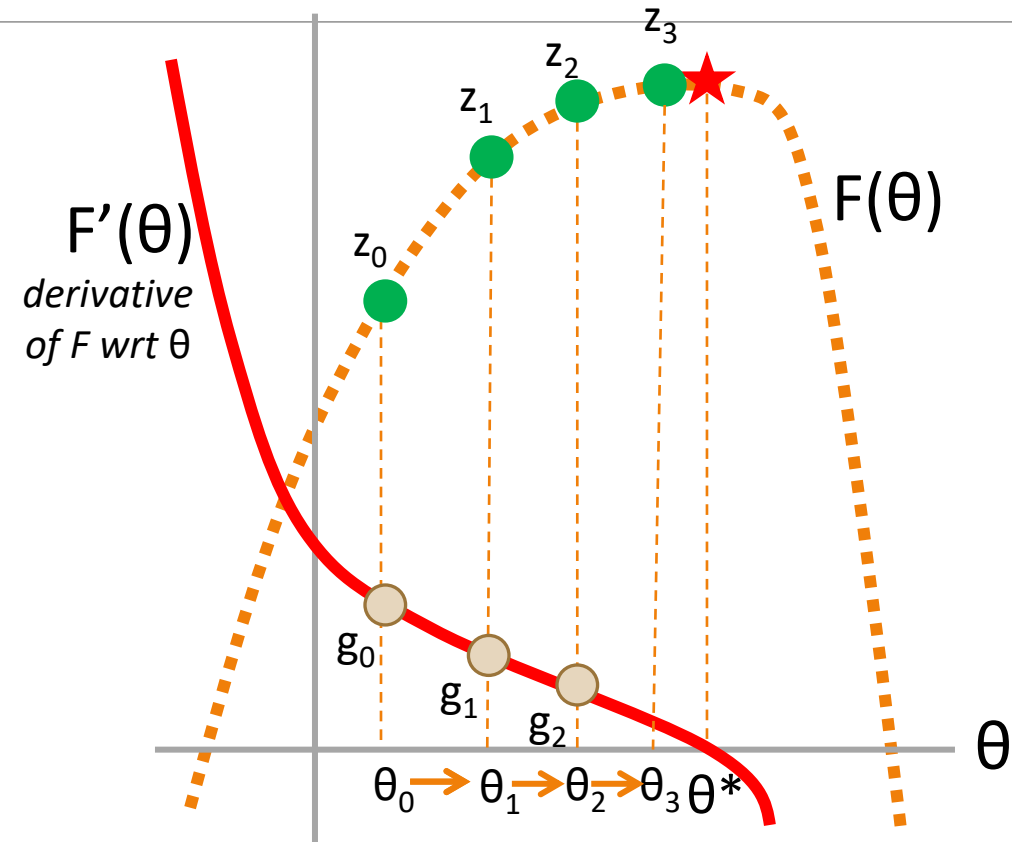
Set $t = 0$

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*



Maxent Models for Classification: Discriminatively or Generatively Trained

Directly model
the posterior

$$p(Y | X) = \text{maxent}(X; Y)$$

Discriminatively trained classifier

Model the
posterior with
Bayes rule

$$p(Y | X) \propto \text{maxent}(X | Y)p(Y)$$

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

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

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

Bayes' Rule \rightarrow Naïve Bayes Assumption

Bayes

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

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

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

Naïve
Bayes

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

Bayes' Rule \rightarrow Naïve Bayes Assumption

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

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

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