CMSC 471: Machine Learning

KMA Solaiman – ksolaima@umbc.edu

Some slides courtesy Tim Finin and Frank Ferraro

Why study learning?

- **Discover** new things or structure previously unknown
  - Examples: data mining, scientific discovery
- Fill in skeletal or **incomplete specifications** in a domain
  - **Large, complex systems** can't be completely built by hand & require dynamic updating to incorporate new info.
  - Learning new characteristics expands the domain or expertise and lessens the "brittleness" of the system
- Acquire models automatically from data rather than by manual programming
- Build agents that can **adapt** to users, other agents, and their environment
- Understand and improve efficiency of **human learning**

What does it mean to learn?

Wesley has been taking an AI course

Geordi, the instructor, needs to determine if Wesley has "learned" the topics covered, at the end of the course

What is a "reasonable" exam?

- (Bad) Choice 1: History of pottery
  - Wesley's performance is not indicative of what was learned in AI
- (Bad) Choice 2: Questions answered during lectures
  - Open book?

A good test should test ability to answer "related" but "new" questions on the exam

Generalization

Model, parameters and hyperparameters

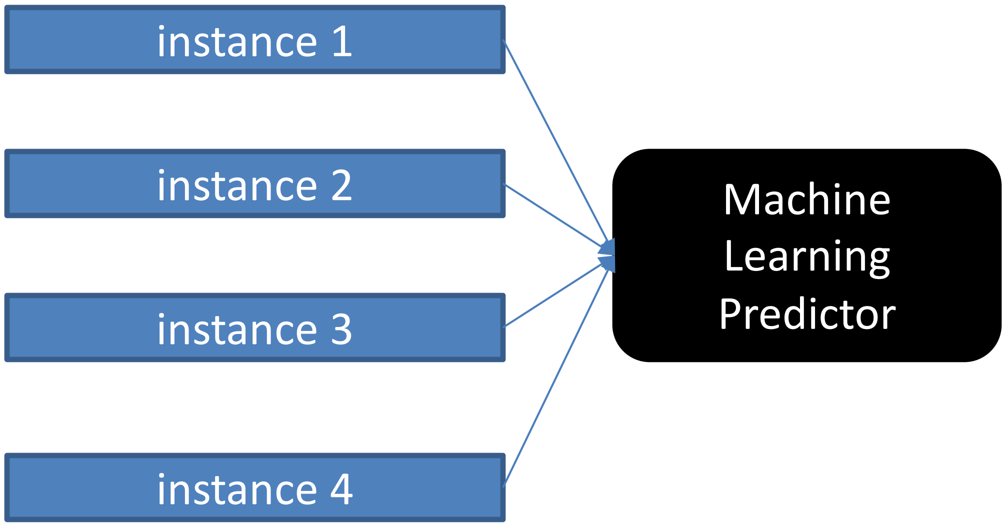
Model: mathematical formulation of system (e.g., classifier)

Parameters: primary "knobs" of the model that are set by a learning algorithm

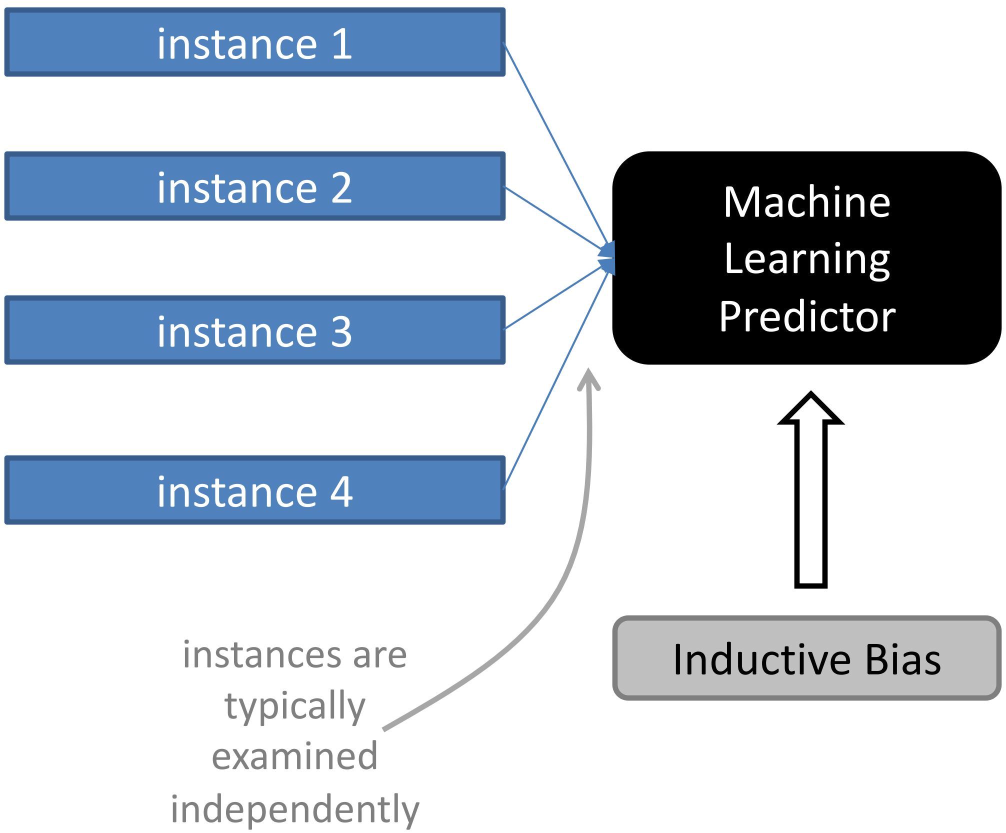
Hyperparameter: secondary "knobs" set by designer

http://www.uiparade.com/wp-content/uploads/2012/01/ui-design-pure-css.jpg

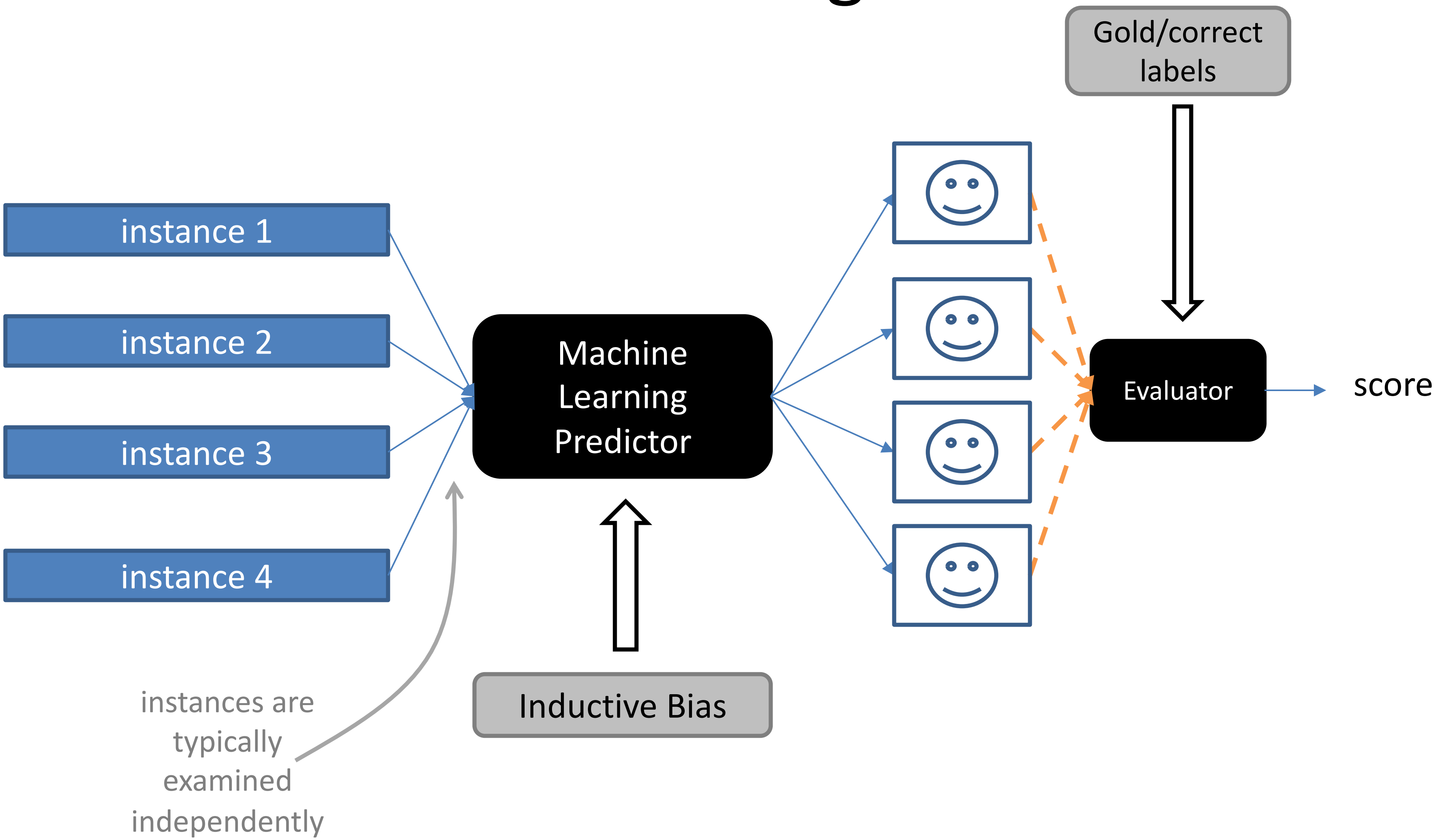
Machine Learning Framework: Learning



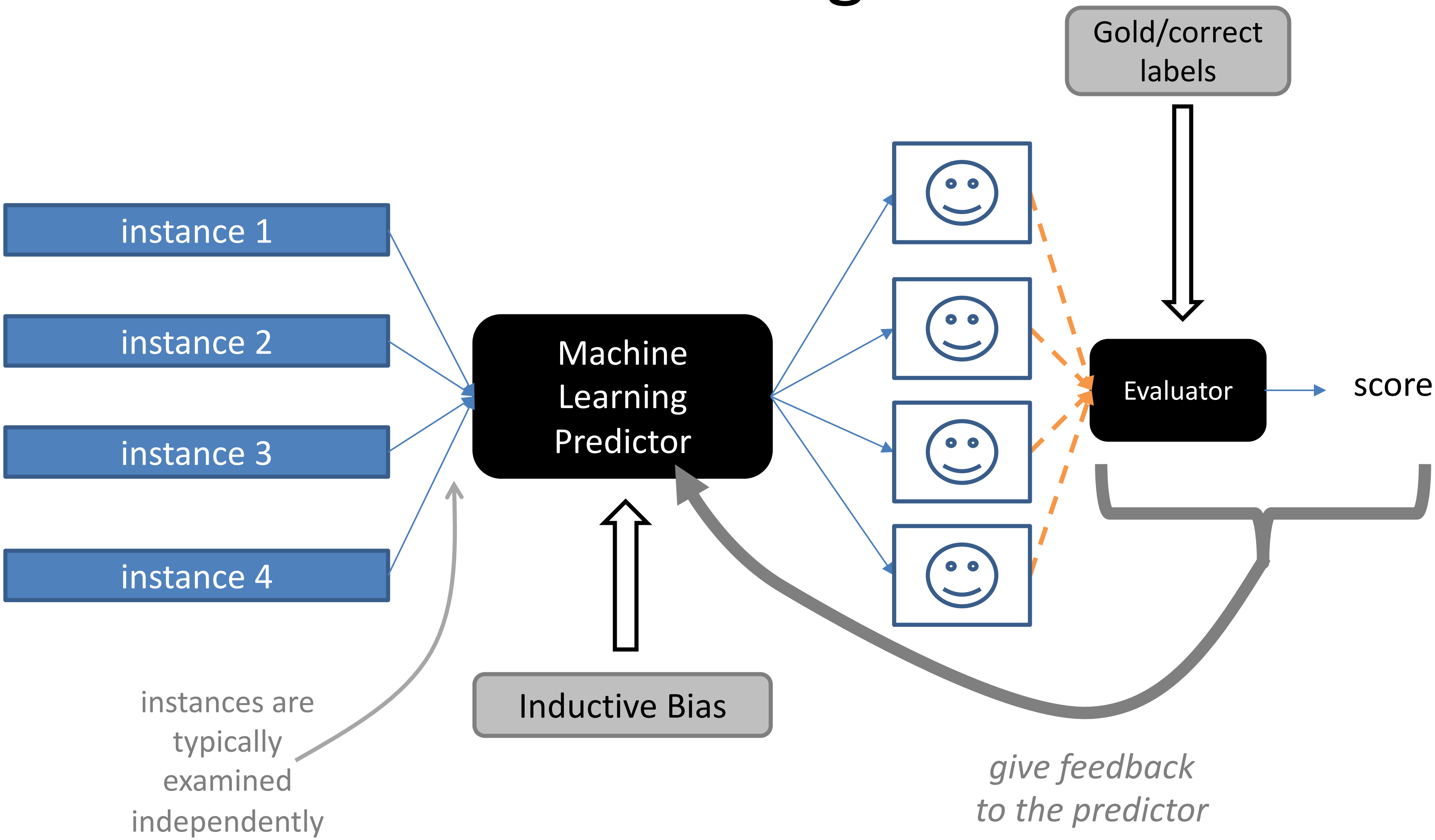
Machine Learning Framework: Learning



Machine Learning Framework: Learning



Machine Learning Framework: Learning

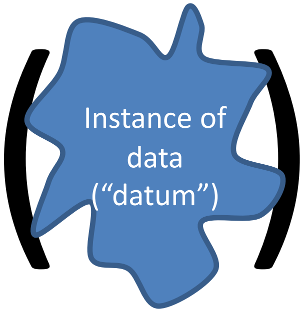


Classify with Goodness

$$\text{predicted label}$$

$$= \underset{\text{label}}{\arg\max}\ \text{score}(\text{example}, \text{label})$$

scoring model

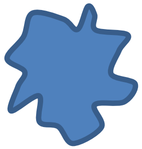
$\text{score}_\theta($ 

$)$

$\Downarrow$

objective

$F(\theta)$

(implicitly) dependent on the observed data X= 

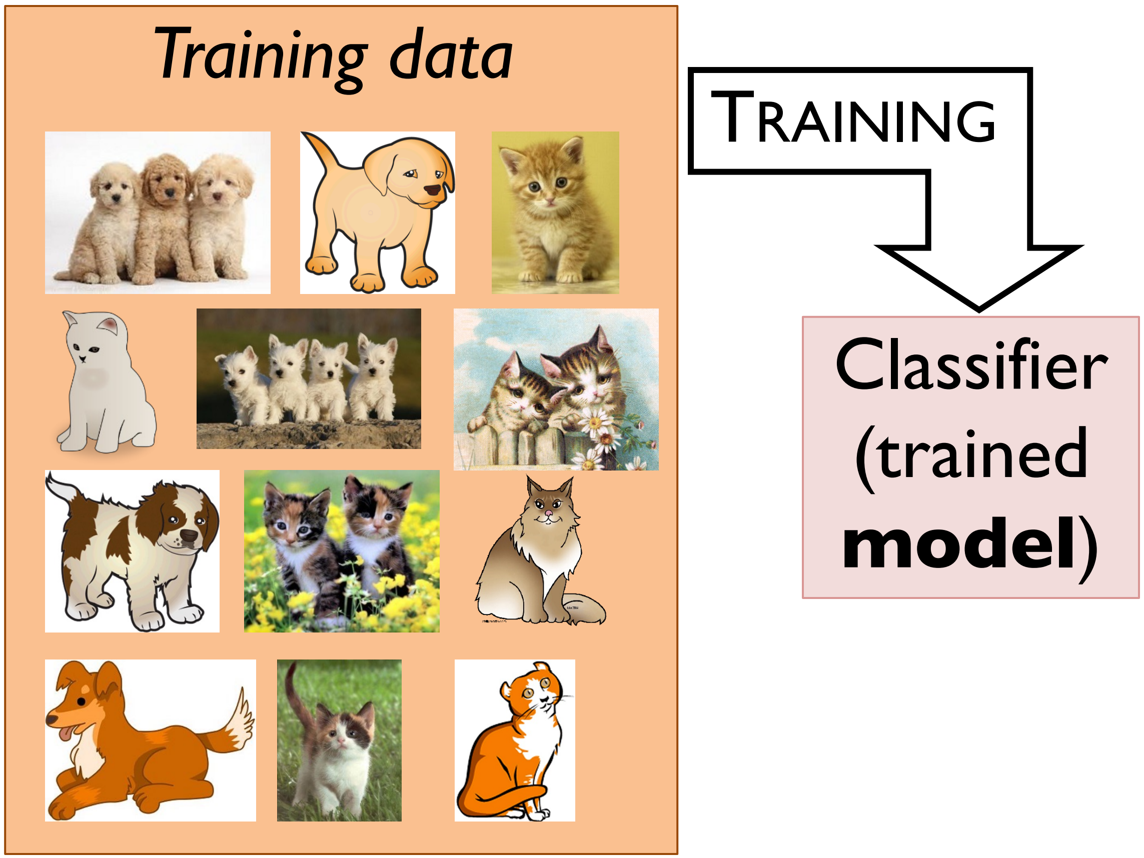
ML Framework Example

Training data

Puppy classifier

ML Framework Example

Puppy classifier



ML Framework Example

Puppy classifier

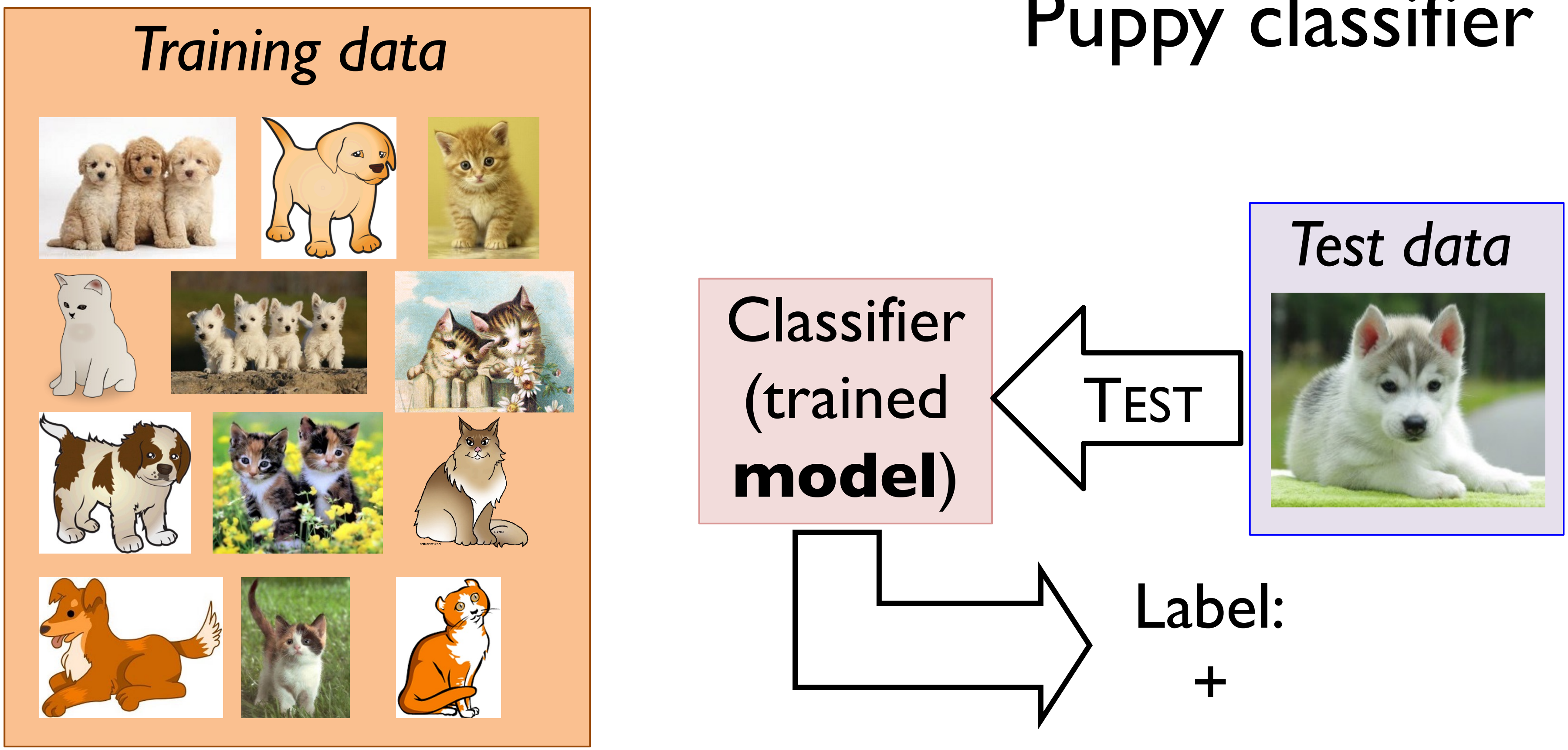
Training data

Classifier (trained **model**)

Test data

ML Framework Example

Puppy classifier

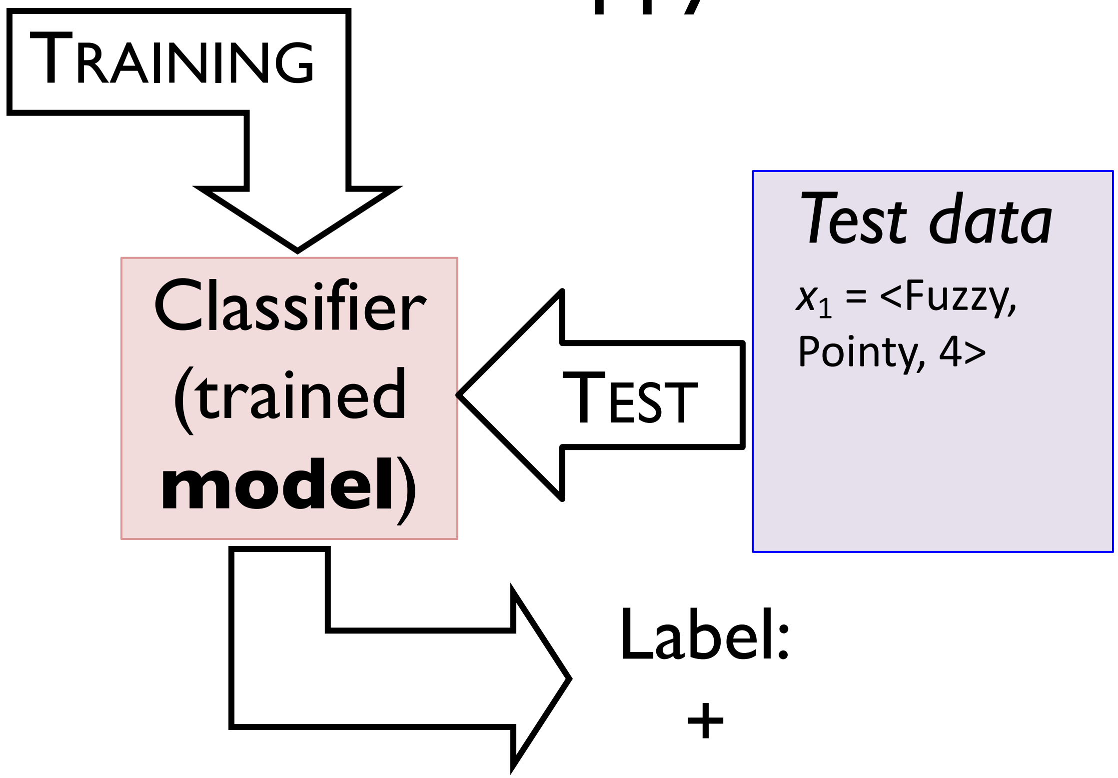


ML Framework Example

Puppy classifier

Training data, X

Text-ure	*Ears*	*Legs*	*Class*
Fuzzy	Round	4	+
Slimy	Missing	8	-
Fuzzy	Pointy	4	-
Fuzzy	Round	4	+
Fuzzy	Pointy	4	+
…			



The Big Idea and Terminology

Given some data, learn a model of how the world works that lets you predict new data

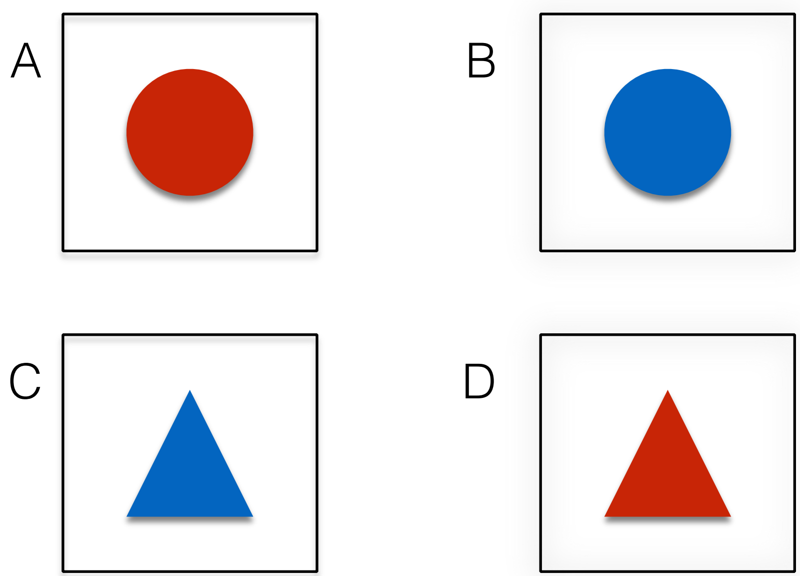
- **Training Set:** Data from which you learn initially
- **Model:** What you learn; a "model" of how inputs are associated with outputs
- **Test set:** New data you test your model against
- **Corpus:** A body of text data (pl.: corpora)
- **Representation:** The computational expression of data

General ML Consideration: Inductive Bias

What do we know *before* we see the data, and how does that influence our modeling decisions?

General ML Consideration: Inductive Bias

What do we know *before* we see the data, and how does that influence our modeling decisions?

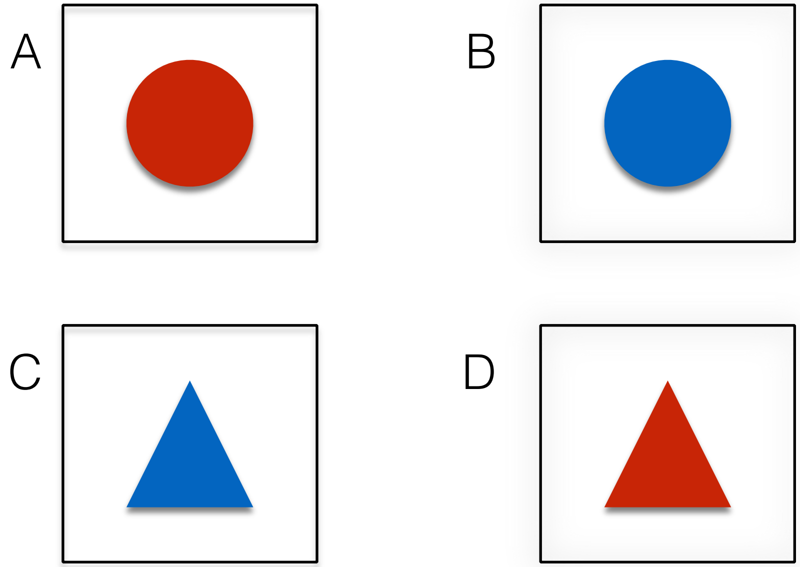


Partition these into two groups…

Courtesy Hamed Pirsiavash

General ML Consideration: Inductive Bias

What do we know *before* we see the data, and how does that influence our modeling decisions?



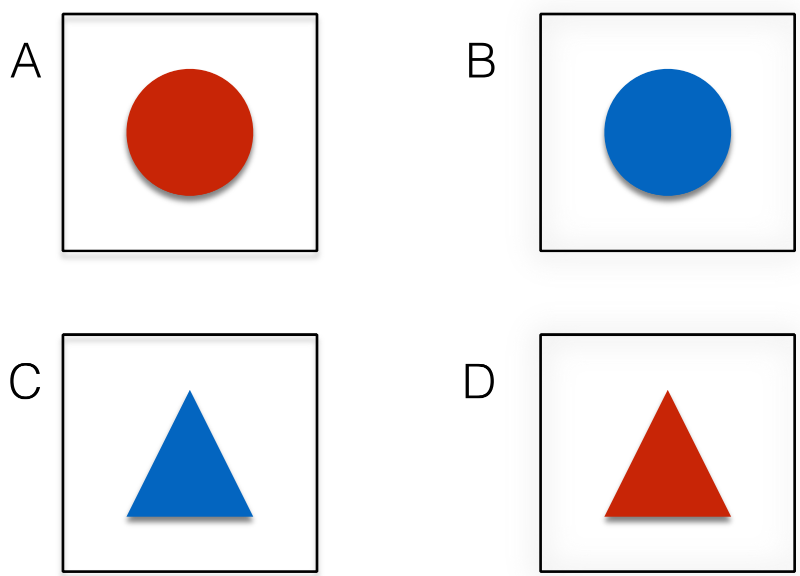
Partition these into two groups

Who selected red vs. blue?

Courtesy Hamed Pirsiavash

General ML Consideration: Inductive Bias

What do we know *before* we see the data, and how does that influence our modeling decisions?



Partition these into two groups

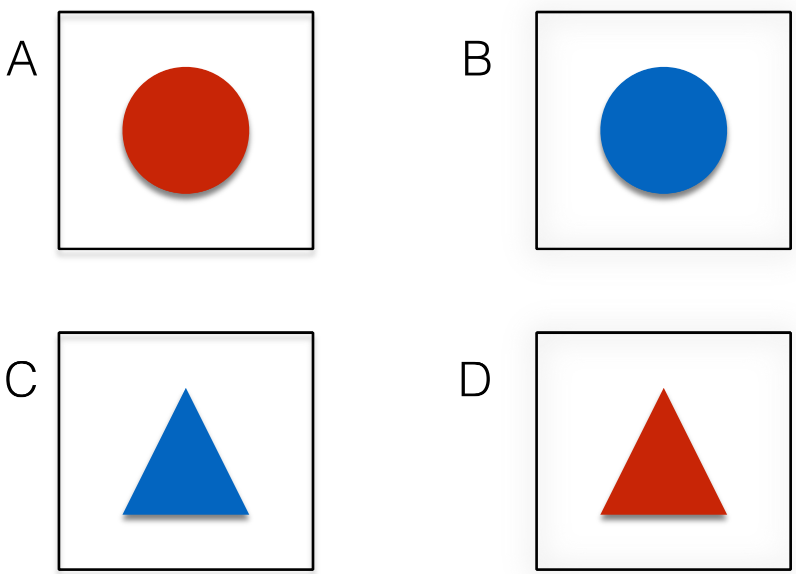
Who selected red vs. blue?

Who selected ● vs. ▲ ?

Courtesy Hamed Pirsiavash

General ML Consideration: Inductive Bias

What do we know *before* we see the data, and how does that influence our modeling decisions?



Partition these into two groups

Who selected red vs. blue?

Who selected ● vs. ▲ ?

Tip: Remember how your own biases/interpretation are influencing your approach

AI & ML

AI and Learning Today

- 50s&60s: neural network learning popular

  Marvin Minsky did neural networks for his dissertation

- Mid 60s: replaced by paradigm of manually encoding & using symbolic knowledge

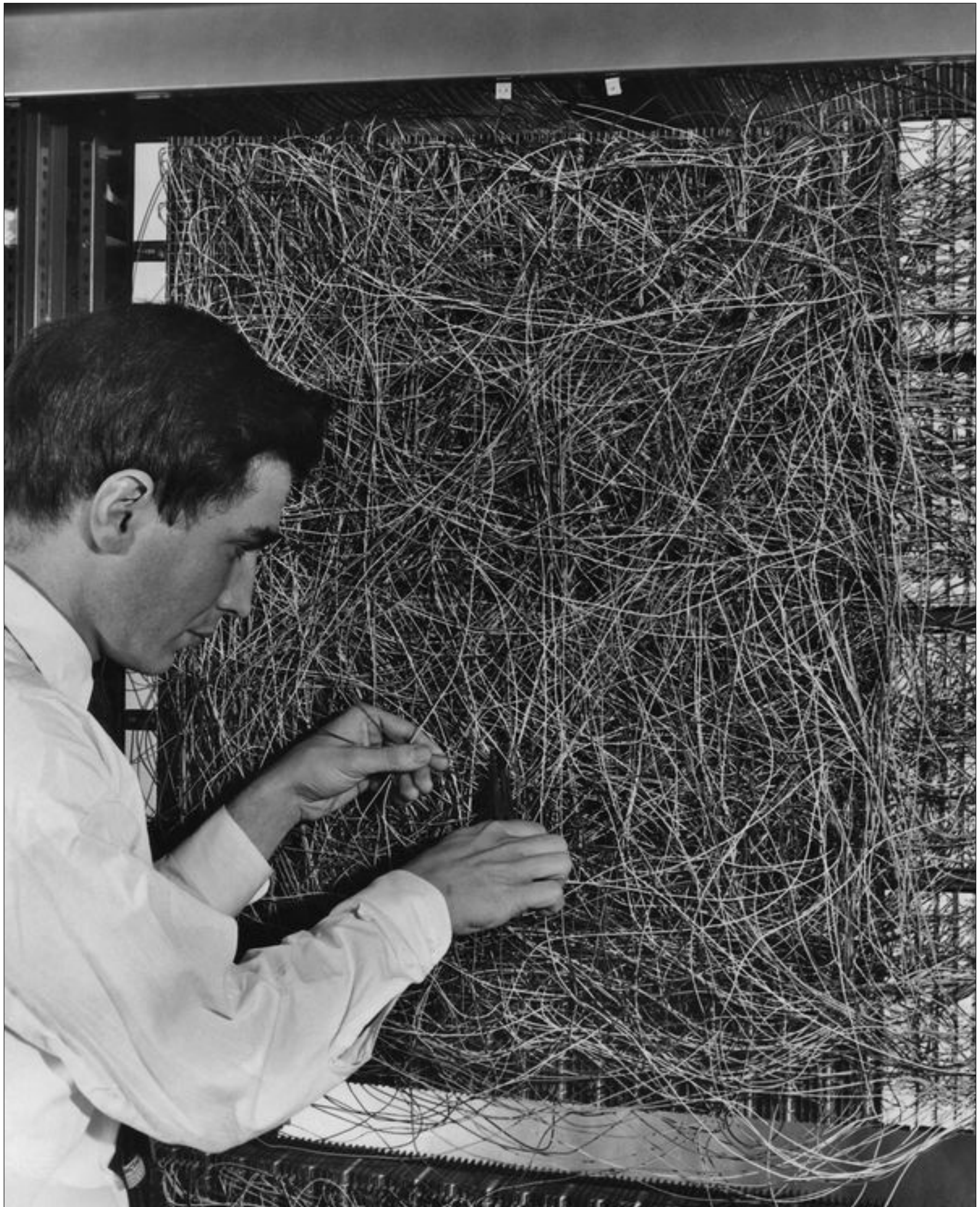
  Cf. Perceptrons, Minsky & Papert book showed limitations of perceptron model of neural networks

- 90s: more data & Web drove interest in statistical machine learning techniques & data mining

- Now: machine learning techniques & big data play biggest driver in almost all successful AI systems

  … and neural networks are the current favorite approach

seeAlso: Timeline of machine learning



Neural Networks 1960

A man adjusting the random wiring network between the light sensors and association unit of scientist Frank Rosen-blatt's Perceptron, or MARK 1 computer, at the Cornell Aeronautical Laboratory, Buffalo, New York, circa 1960. The machine is designed to use a type of artificial neural network, known as a perceptron.

Machine Learning Successes

- Games: chess, go, poker
- Text sentiment analysis
- Email spam detection
- Recommender systems (e.g., Netflix, Amazon)
- Machine translation
- Speech understanding
- SIRI, Alexa, Google Assistant, …

- Autonomous vehicles
- Individual face recognition
- Understanding digital images
- Credit card fraud detection
- Showing annoying ads

Not Done yet

Major Machine learning paradigms (1)

- **Rote**: 1-1 mapping from inputs to stored representation, learning by memorization, association-based storage & retrieval
- **Induction:** Use specific examples to reach general conclusions
- **Clustering**: Unsupervised discovery of natural groups in data

Major Machine learning paradigms (2)

- **Analogy:** Find correspondence between different representations
- **Discovery**: Unsupervised, specific goal not given
- **Genetic algorithms:** *Evolutionary* search techniques, based on *survival of the fittest*
- **Reinforcement:** Feedback (positive or negative reward) given at the end of a sequence of steps
- **Deep learning:** *artificial neural networks* with *representation learning* for ML tasks

TYPES OF LEARNING

Three Axes for Thinking About Your ML Problem

Classification	Fully-supervised	Probabilistic	Neural
Regression	Semi-supervised	Generative	Memory-based
Clustering	Un-supervised	Conditional	Exemplar
		Spectral	…

*the **task**: what* kind *of problem are you solving?*

*the **data**: amount of human input/number of labeled examples*

*the **approach**: how any data are being used*

Types of learning problems

- **Supervised**: learn from training examples
  - Regression:
  - Classification: Decision Trees, SVM
- **Unsupervised**: learn w/o training examples
  - Clustering
  - Dimensionality reduction
  - Word embeddings
- **Reinforcement learning:** improve performance using feedback from actions taken
- **Lots more we won't cover**
  - Hidden Markov models, Learning to rank, Semi-supervised learning, Active learning …

Machine Learning Problems

	Supervised Learning	***Unsupervised Learning***
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Inductive Learning Framework

- Raw input data from sensors or a database preprocessed to obtain **feature vector**, **X**, of **relevant** features for classifying examples
- Each **X** is a list of (attribute, value) pairs
- ***n*** attributes (a.k.a. features): fixed, positive, and finite
- Features have fixed, finite number # of possible values
  - Or continuous within some well-defined space, e.g., “age”
- Each example is a point in an *n*-dimensional feature space
  - X = [Person:Sue, EyeColor:Brown, Age:Young, Sex:Female]
  - X = [Cheese:*f*, Sauce:*t*, Bread:*t*]
  - X = [Texture:Fuzzy, Ears:Pointy, Purrs:Yes, Legs:4]

SUPERVISED LEARNING

Supervised learning

- Given training examples of inputs & corres-ponding outputs, produce "correct" outputs for new inputs
- Two important scenarios:
  - **Classification:** outputs typically labels (goodRisk, badRisk); learn decision boundary to separate classes
  - **Regression:** aka *curve fitting* or *function approximation*; Learn a *continuous* input-output mapping from examples, e.g., for a zip code, predict house sale price given its square footage

Supervised learning

- Given training examples of inputs & corres-ponding outputs, produce "correct" outputs for new inputs
- Two important scenarios:
  - **Classification:** outputs typically labels (goodRisk, badRisk); learn decision boundary to separate classes
  - **Regression:** aka *curve fitting* or *function approximation*; Learn a *continuous* input-output mapping from examples, e.g., for a zip code, predict house sale price given its square footage

Classification Examples

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

…

Classification Examples

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

…

Input:

an instance

a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

Output: a predicted class c from C

Classification: Hand-coded Rules?

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

…

Rules based on combinations of words or other features

spam: black-list-address OR ("dollars" AND "have been selected")

Accuracy can be high

If rules carefully refined by expert

Building and maintaining these rules is expensive

Can humans faithfully assign uncertainty?

Classification: Supervised Machine Learning

- Assigning subject categories, topics, or genres
- Spam detection
- Authorship identification
- Age/gender identification
- Language Identification
- Sentiment analysis
- …

Input:

an instance d

a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

A training set of m hand-labeled instances $(d_1,c_1),\ldots,(d_m,c_m)$

Output:

a learned classifier γ that maps instances to classes

Classification: Supervised Machine Learning

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

…

Input:

an instance d

a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

A training set of m hand-labeled instances $(d_1,c_1),\ldots,(d_m,c_m)$

Output:

a learned classifier γ that maps instances to classes

γ learns to associate certain ***features*** of instances with their labels

Classification: Supervised Machine Learning

Assigning subject categories, topics, or genres

Spam detection

Authorship identification

Age/gender identification

Language Identification

Sentiment analysis

…

Input:

an instance d

a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

A training set of m hand-labeled instances $(d_1,c_1), \ldots, (d_m,c_m)$

Output:

a learned classifier γ that maps instances to classes

Naïve Bayes

Logistic regression

Support-vector machines

k-Nearest Neighbors

…

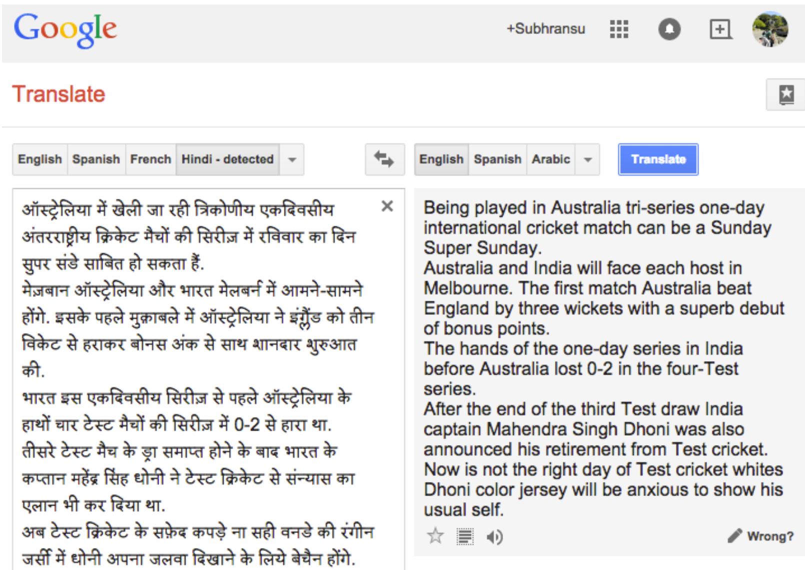
Classification Example: Face Recognition

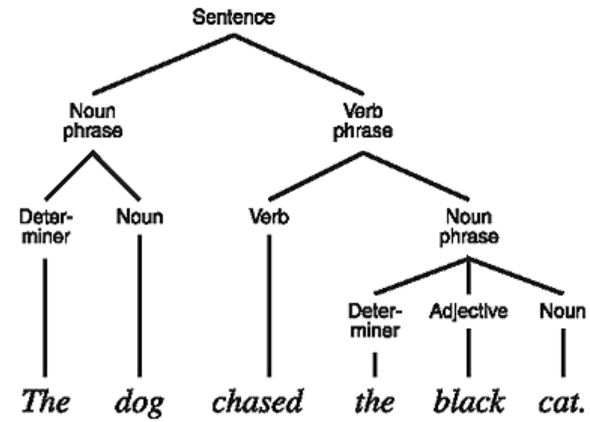
What is a good *representation* for images?

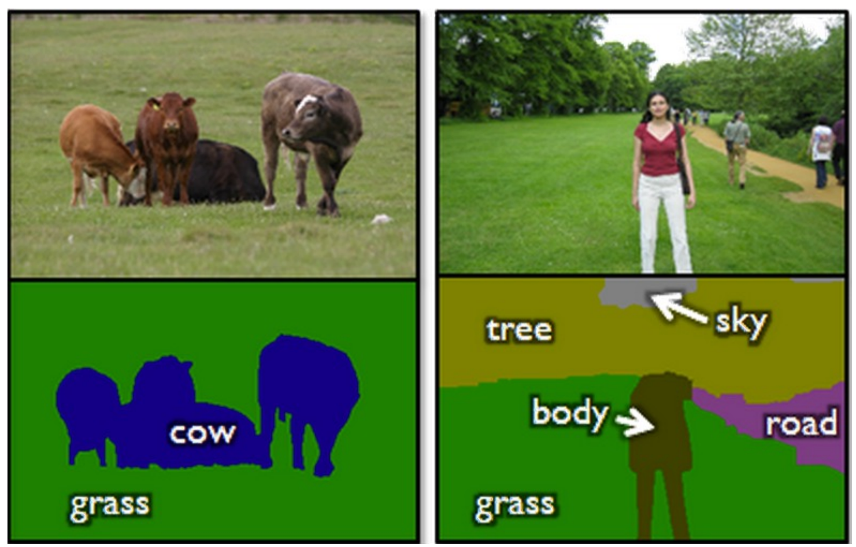
Pixel values? Edges?

Courtesy from Hamed Pirsiavash

Classification Example:
Sequence & Structured Prediction







Courtesy Hamed Pirsiavash

Ingredients for classification

Inject *your* knowledge into a learning system

Feature representation

Training data:
labeled examples

Model

Ingredients for classification

Inject *your* knowledge into a learning system

Problem specific

Difficult to learn from bad ones

Feature representation

Training data:
labeled examples

Model

Courtesy Hamed Pirsiavash

Ingredients for classification

Inject *your* knowledge into a learning system

Problem specific

Difficult to learn from bad ones

Labeling data == $$$

Sometimes data is available for "free"

Feature representation

Training data: labeled examples

Model

Courtesy Hamed Pirsiavash

Ingredients for classification

Inject *your* knowledge into a learning system

Problem specific

Difficult to learn from bad ones

Feature representation

Labeling data == $$$

Sometimes data is available for "free"

Training data: labeled examples

No single learning algorithm is always good ("no free lunch")

Different learning algorithms work differently

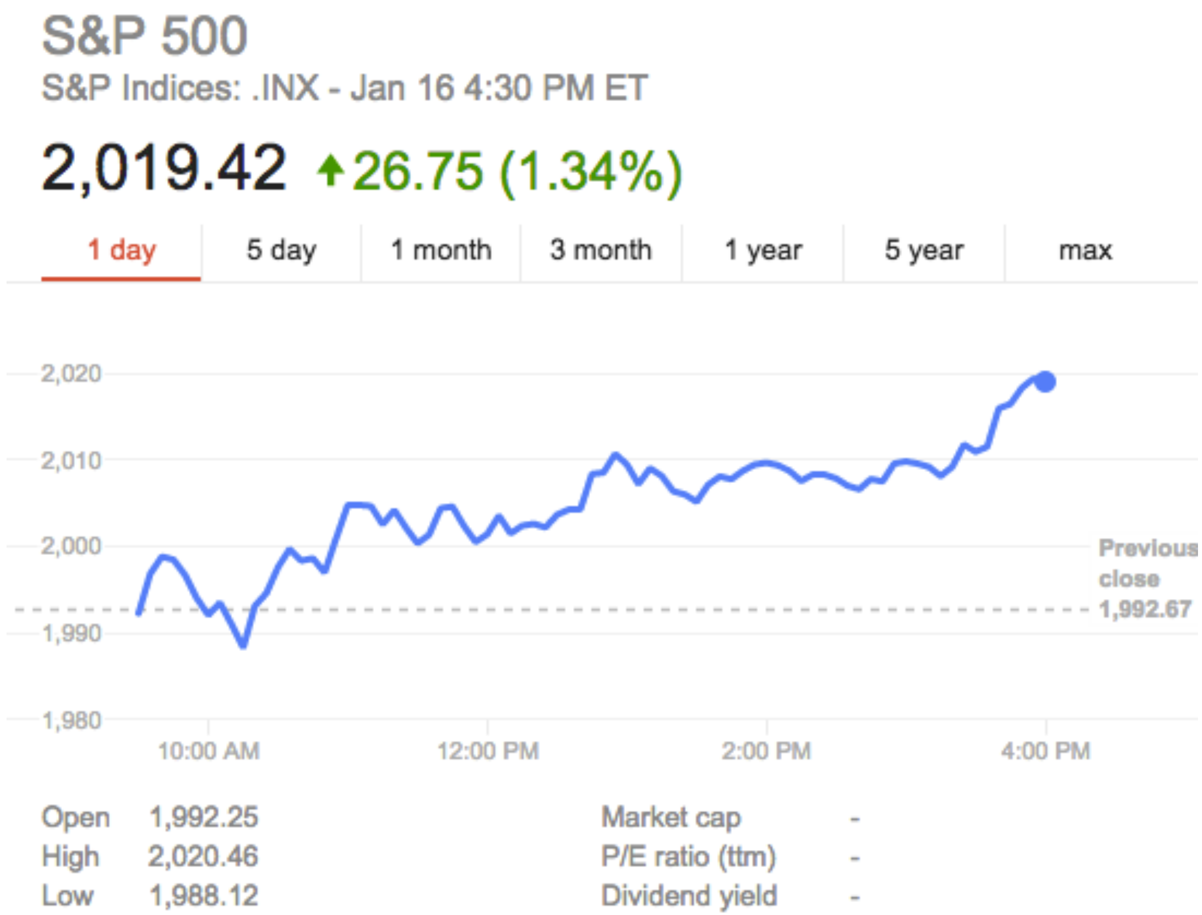
Model

Courtesy Hamed Pirsiavash

Regression

Like classification, but real-valued

Regression Example: Stock Market Prediction



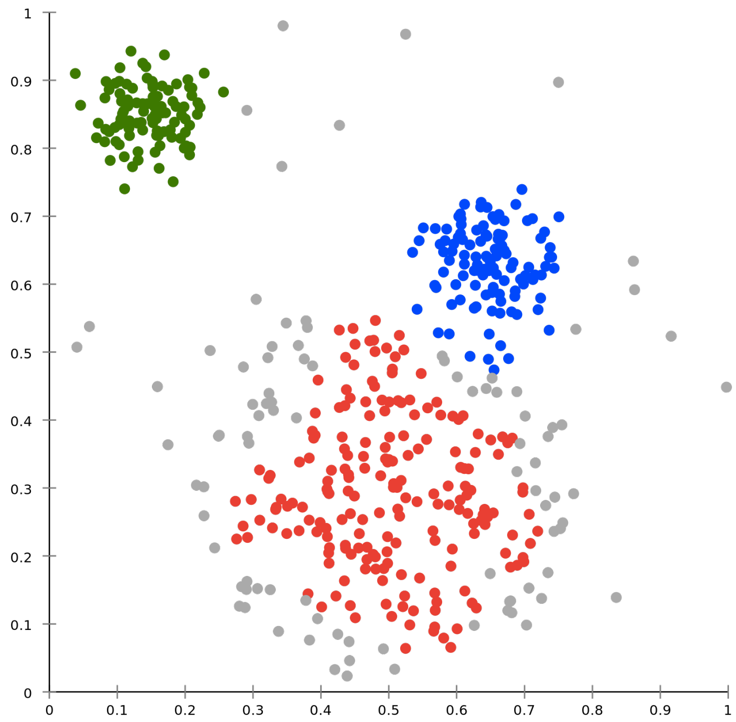
Courtesy Hamed Pirsiavash

Unsupervised Learning

Given only *unlabeled* data as input, learn some sort of structure, e.g.:

- **Clustering**: group Facebook friends based on similarity of post texts and friends
- **Embeddings**: Find sets of words whose meanings are related (e.g., doctor, hospital)
- **Topic modelling**: Induce N topics and words most common in documents about each

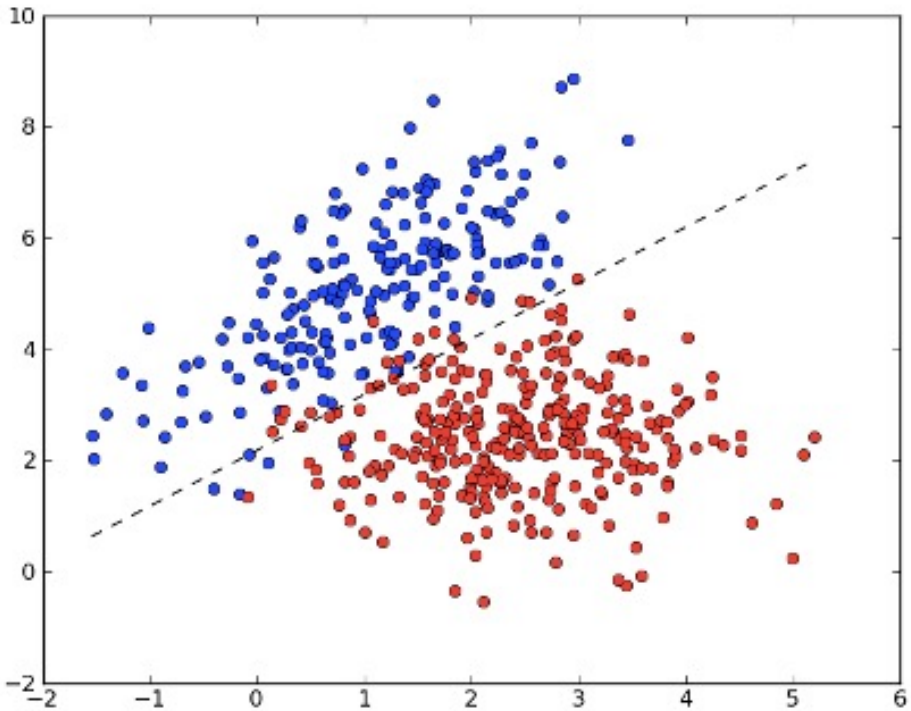
Unsupervised learning: Clustering



Courtesy Hamed Pirsiavash

LINEAR MODELS

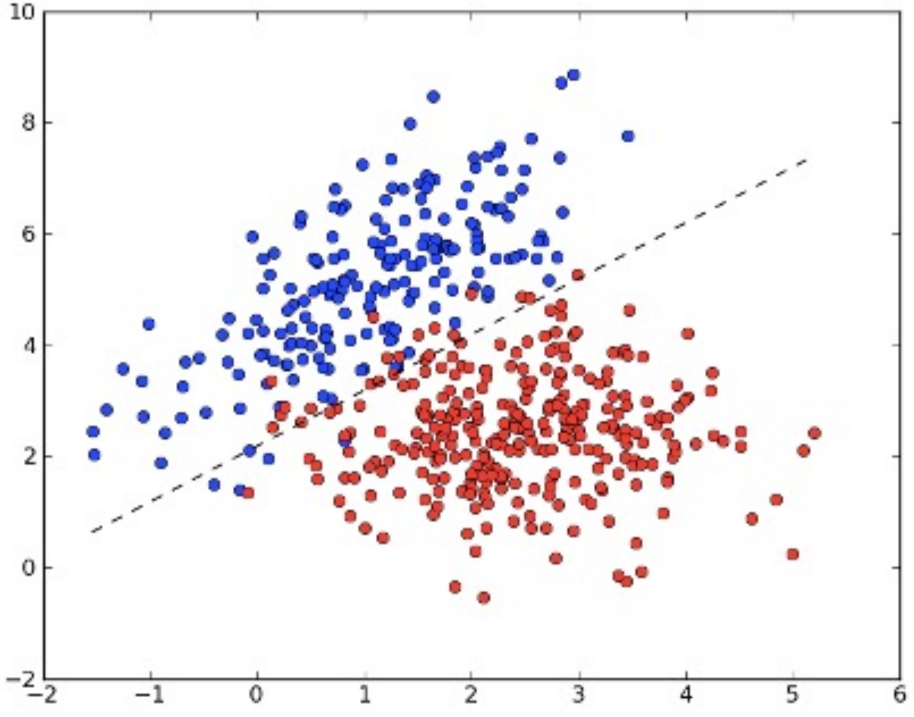
Linear Models



- Can be used for either regression or classification
- A number of instances for classification. Common ones are:
  – Perceptron
  – Linear SVM
  – Logistic regression
    - (yes, even though "regression" is in the name ☺)

Image courtesy Hamed Pirsiavash

Linear Models: Core Idea



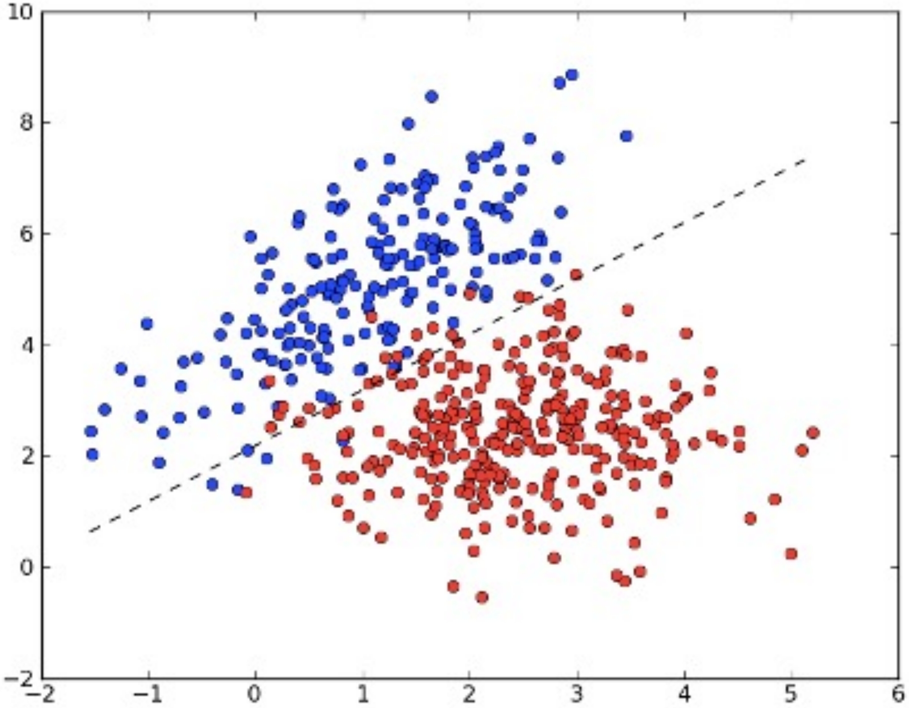
Model the relationship between the input data X and corresponding labels Y via a linear relationship (non-zero intercepts b are okay)

$$Y = W^T X + b$$

Items to learn: W, b

Image courtesy Hamed Pirsiavash

Linear Models: Core Idea



Model the relationship between the input data X and corresponding labels Y via a linear relationship (non-zero intercepts b are okay)

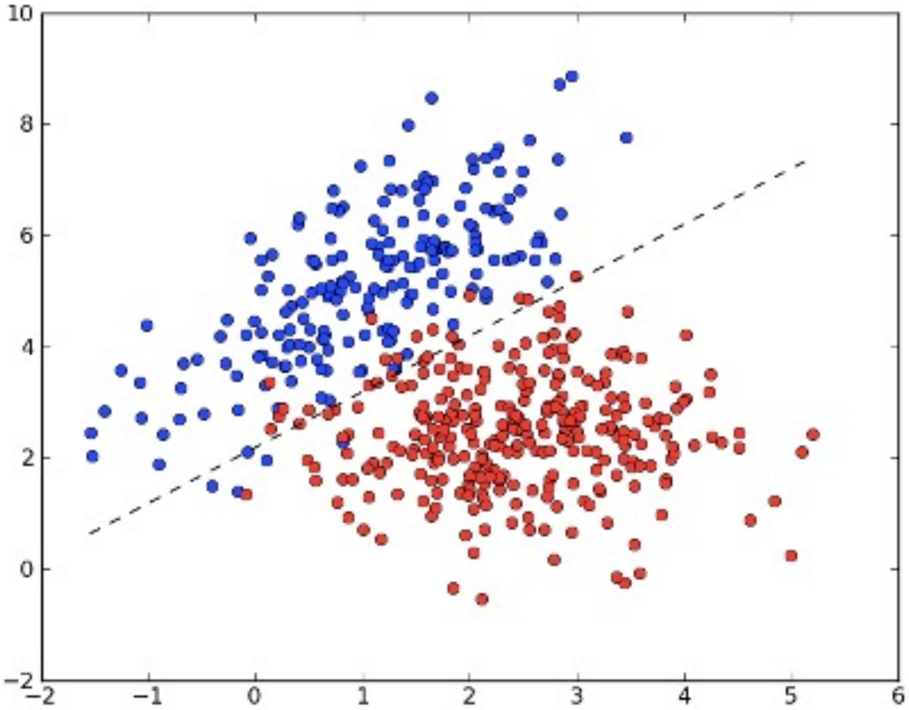
$$Y = W^T X + b$$

Items to learn: W, b

For regression: the output of this equation *is* the predicted value

Image courtesy Hamed Pirsiavash

Linear Models: Core Idea



Model the relationship between the input data X and corresponding labels Y via a linear relationship (non-zero intercepts b are okay)

$$Y = W^T X + b$$

Items to learn: W, b

For regression: the output of this equation *is* the predicted value

For classification: one class is on one side of this line, the other class is on the other

Image courtesy Hamed Pirsiavash

Linear Models in sklearn

These all have easy-to-use interfaces, with the same core interface:

- Training:

$$model.fit(X = training\_features, y = training\_labels)$$

- Prediction:

$$model.predict(X = eval\_features)$$

https://scikit-learn.org/stable/modules/linear_model.html

Linear Models in pytorch

Docs > torch.nn > Linear

LINEAR

CLASS `torch.nn.Linear(in_features, out_features, bias=True)`

Applies a linear transformation to the incoming data: $y = xA^T + b$

This module supports TensorFloat32.

Variables

- **~Linear.weight** – the learnable weights of the module of shape $(\text{out\_features}, \text{in\_features})$. The values are initialized from $\mathcal{U}(-\sqrt{k}, \sqrt{k})$, where $k = \frac{1}{\text{in\_features}}$
- **~Linear.bias** – the learnable bias of the module of shape (out_features). If `bias` is `True`, the values are initialized from $\mathcal{U}(-\sqrt{k}, \sqrt{k})$ where $k = \frac{1}{\text{in\_features}}$

Examples:

```
>>> m = nn.Linear(20, 30)
>>> input = torch.randn(128, 20)
>>> output = m(input)
>>> print(output.size())
torch.Size([128, 30])
```

These are "building blocks" not full models.

https://pytorch.org/docs/stable/generated/torch.nn.Linear.html

A Simple Linear Model

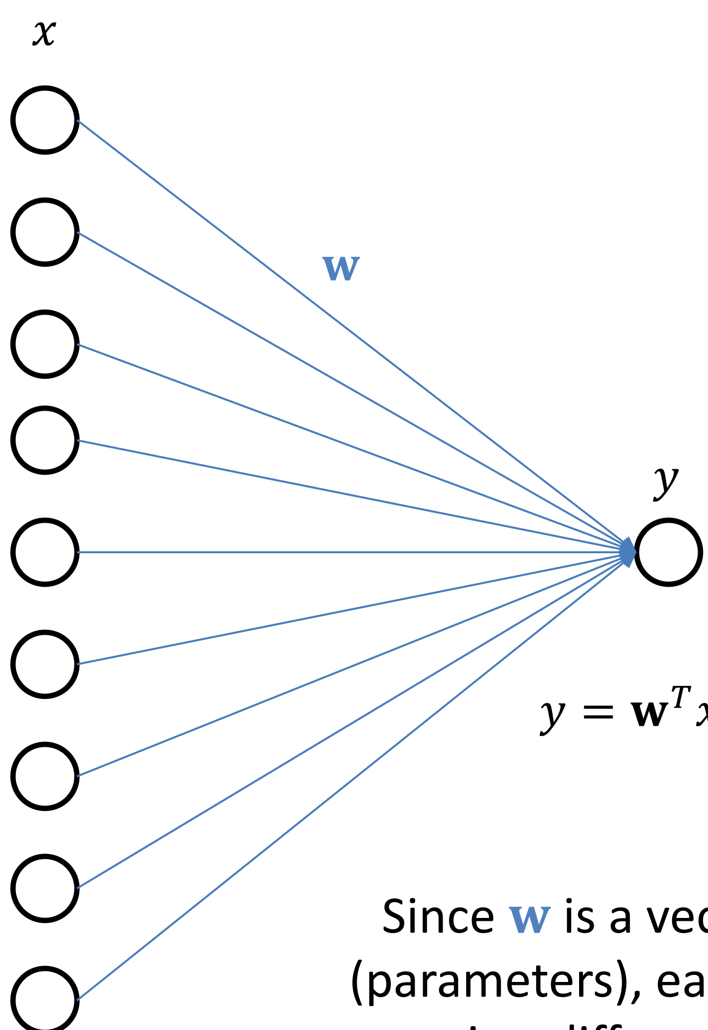
Take CMSC 478 (or 678), or independent study to learn about this in more detail!

$$\text{predict } y_i \text{ from } \mathbf{x_i}$$

value y_i

data point x_i, as a vector of features

A Graphical View of Linear Models



$$y = \mathbf{w}^T x + b$$

Since $\mathbf{w}$ is a vector of weights (parameters), each arc from x to y is a different parameter

A Simple Linear Model for Regression

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

value y_i

data point x_i, as a vector of features

A Simple Linear Model for Regression

vector w of weights

bias b

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}} + b$$

value y_i

data point x_i, as a vector of features

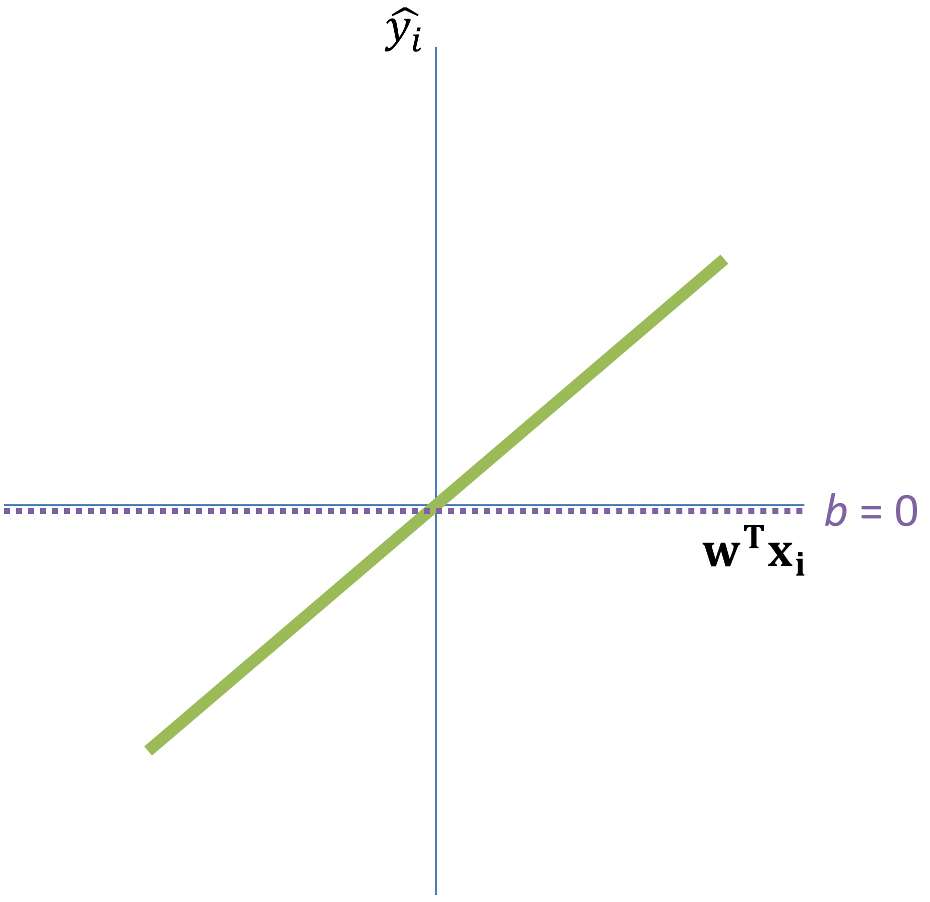
A Simple Linear Model for Regression

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}} + 0$$

value y_i

data point x_i, as a vector of features



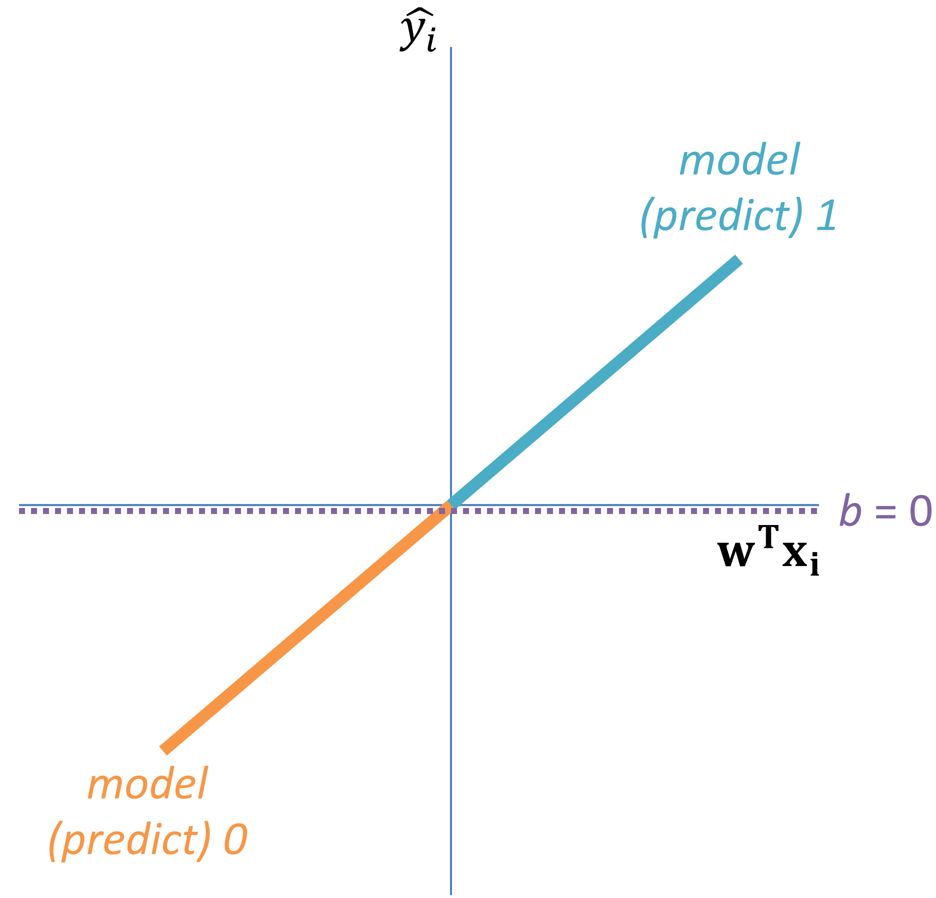
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^\mathbf{T}\mathbf{x_i}$$

label y_i (binary {0, 1} value)

data point x_i, as a vector of features



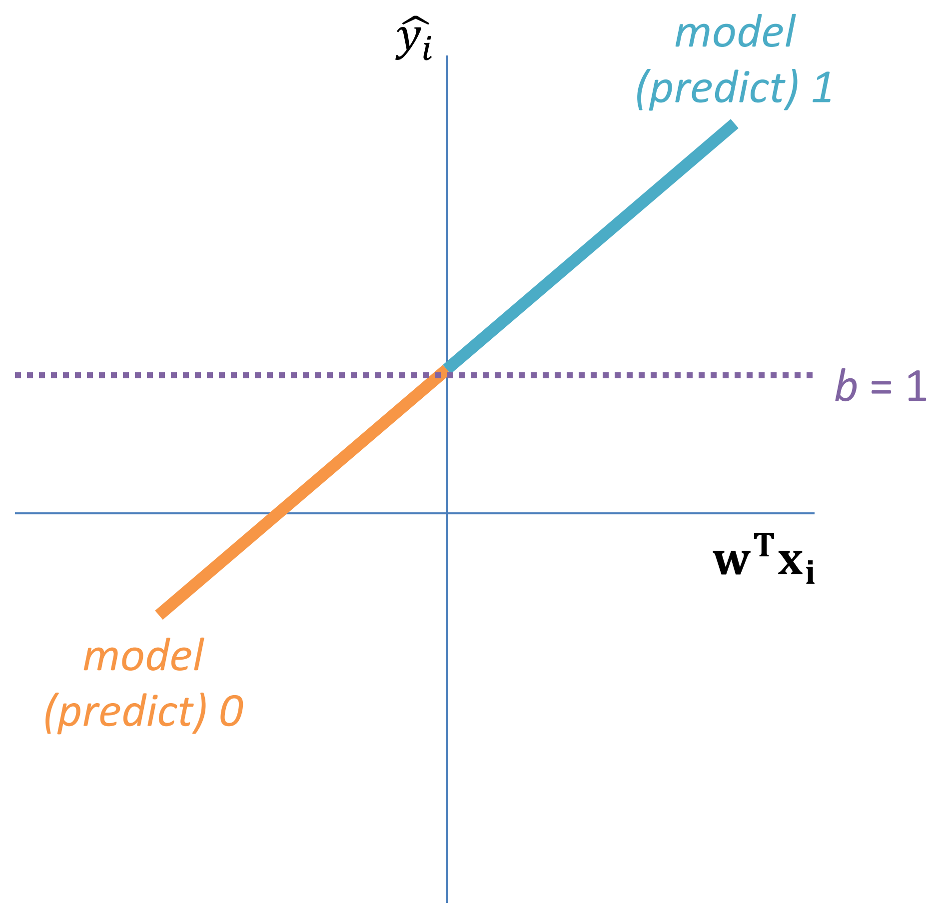
A Simple Linear Model for Classification

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x}_{\mathbf{i}}+1$$

label y_i (binary {0, 1} value)

data point x_i, as a vector of features



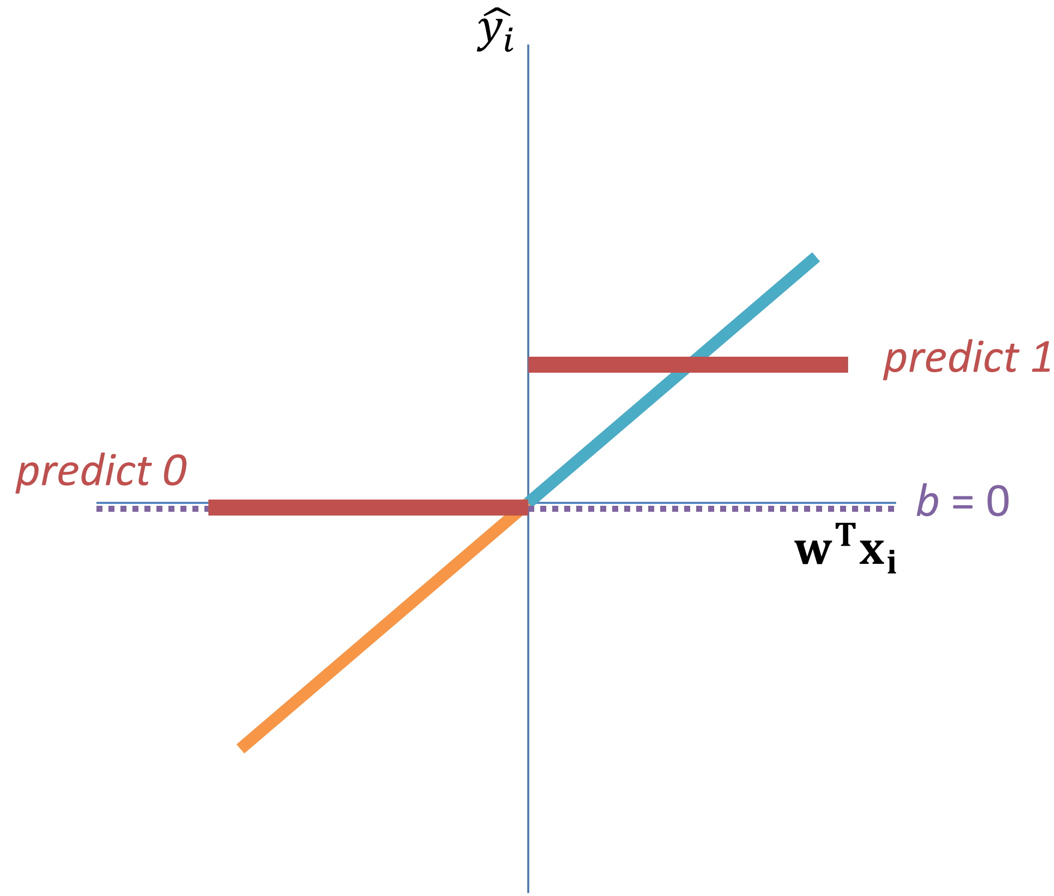
Decision Rules

vector w of weights

$$y_i = \mathbf{w}^{\mathbf{T}}\mathbf{x_i}$$

label y_i (binary {0, 1} value)

data point x_i, as a vector of features



decision rule: $\widehat{y}_i = \begin{cases} 0, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} < 0 \\ 1, & \mathbf{w}^{\mathbf{T}}\mathbf{x_i} \geq 0 \end{cases}$

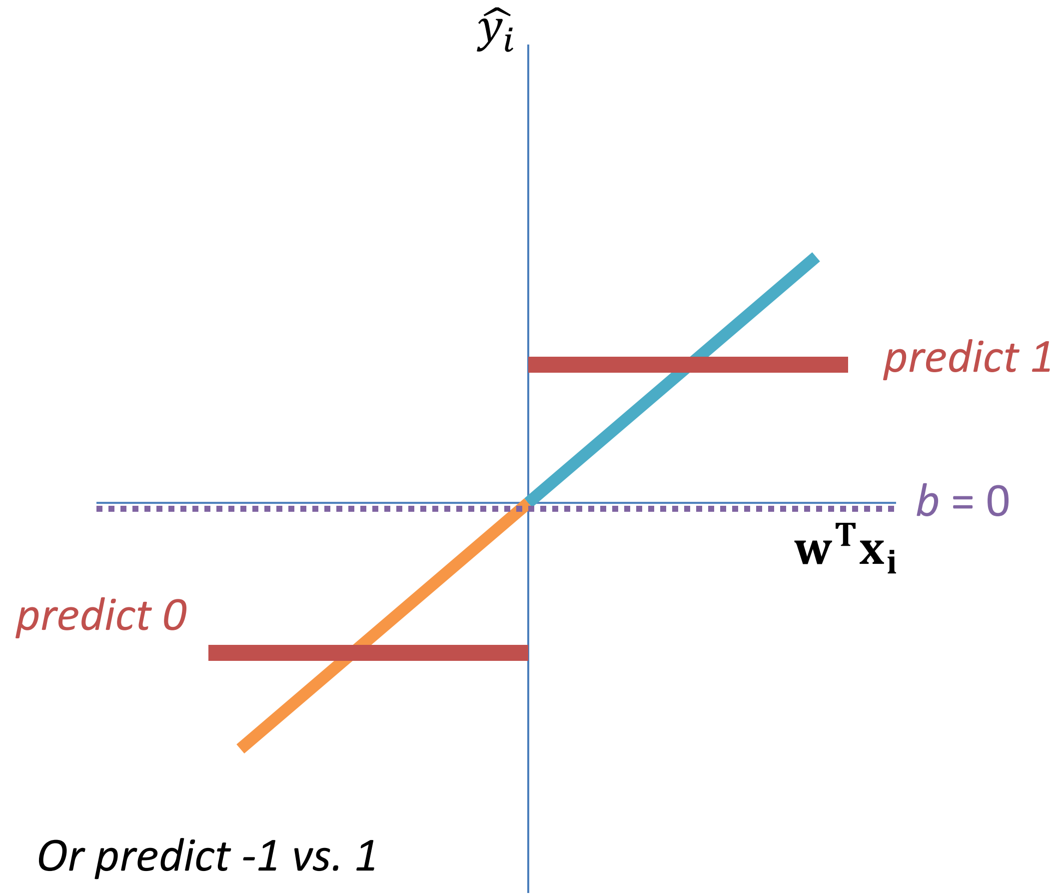
Decision Rules

vector w of weights

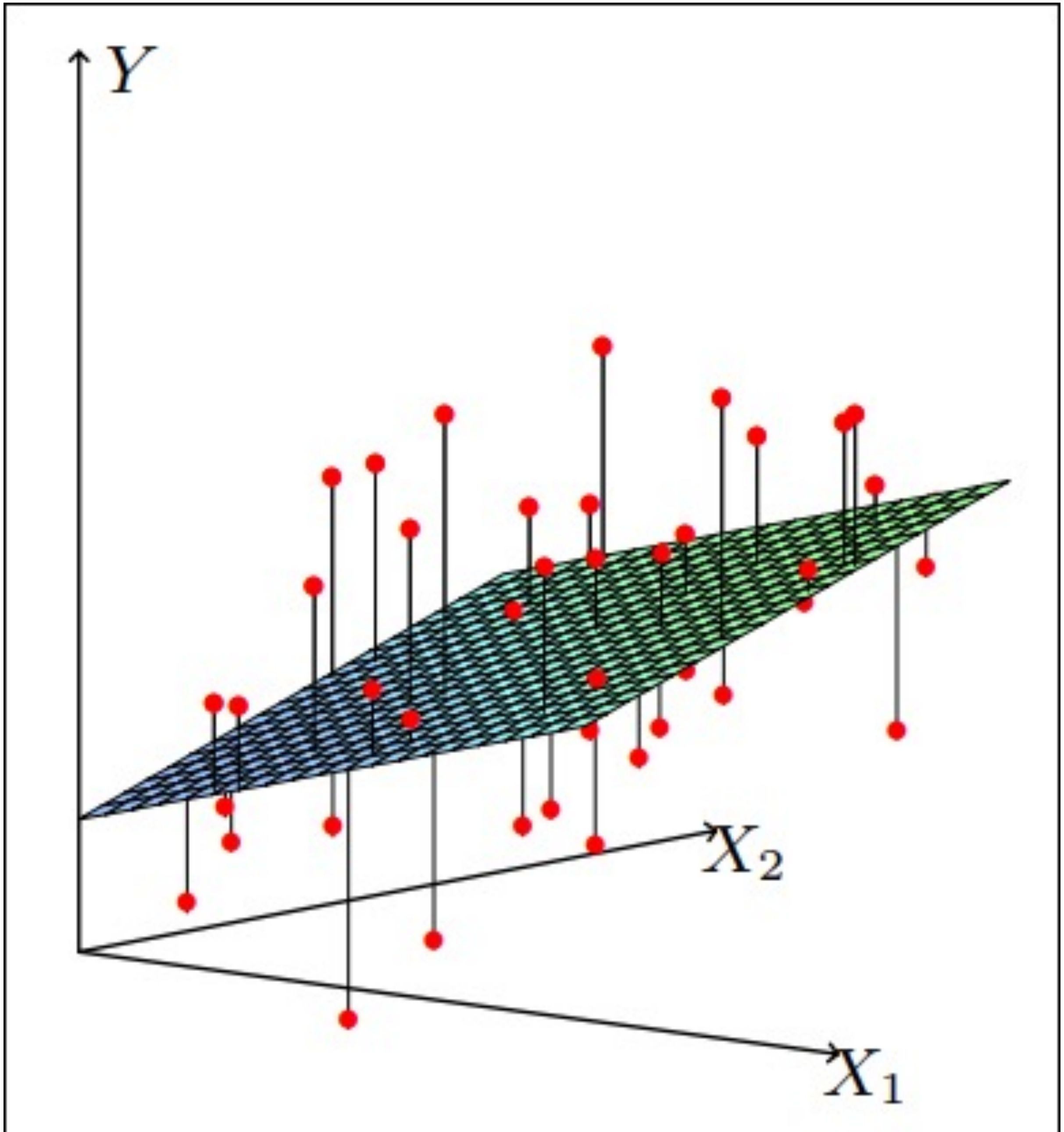
$$y_i = \mathbf{w}^\mathbf{T}\mathbf{x_i}$$

label y_i (binary {0, 1} value)

data point x_i, as a vector of features



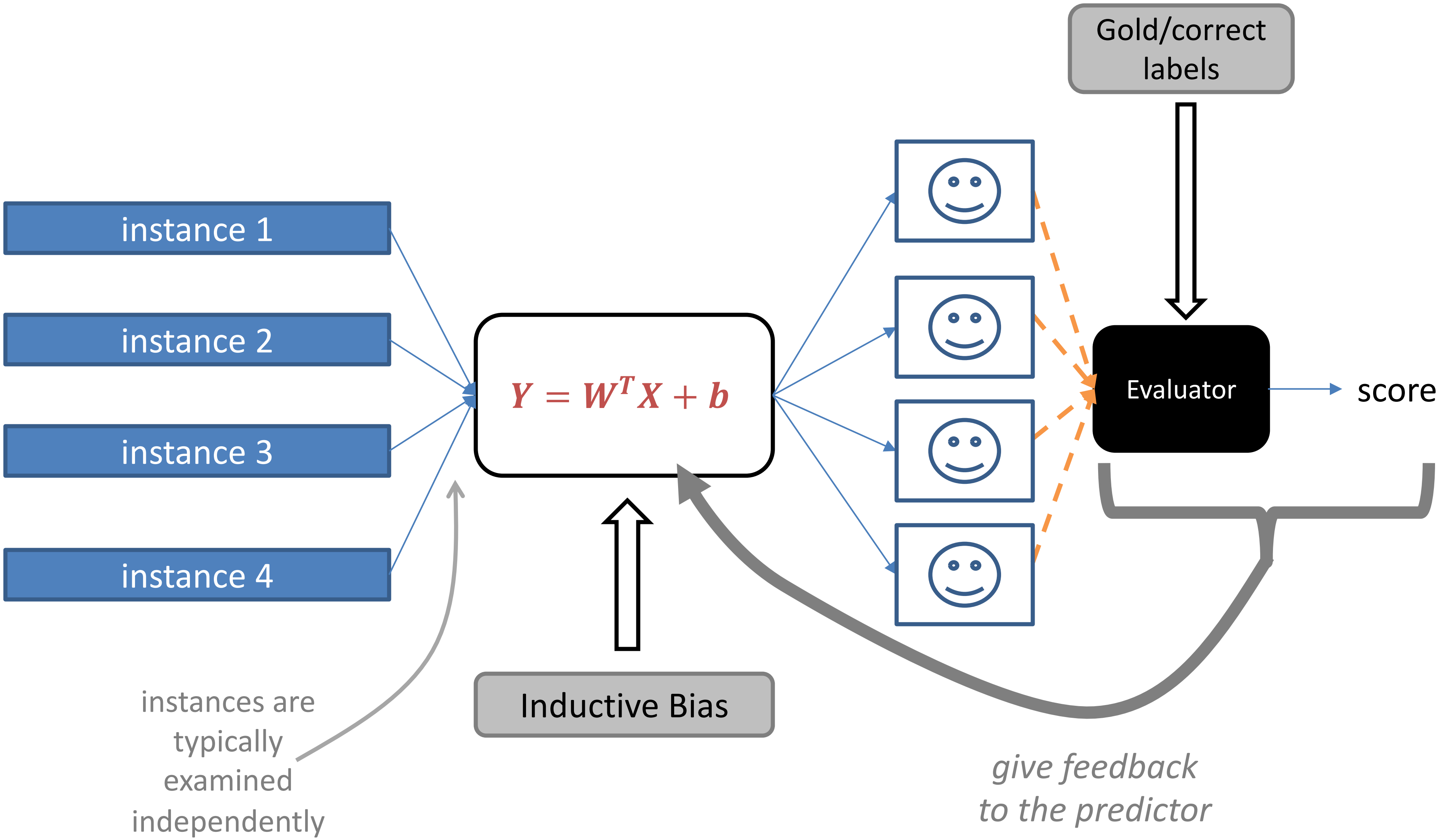
decision rule: $\hat{y}_i = \begin{cases} -1, & \mathbf{w}^\mathbf{T}\mathbf{x_i} < 0 \\ 1, & \mathbf{w}^\mathbf{T}\mathbf{x_i} \geq 0 \end{cases}$



Linear Models in Multiple Dimensions

ESL, Fig 3.1

Linear Models in the Basic Framework



Central Question: How Well Are We Doing?

Reminder!

The performance score does not have to be the same thing as the loss function you optimize

Classification

- Precision, Recall, F1
- Accuracy
- Log-loss
- ROC-AUC
- …

Regression

- (Root) Mean Square Error
- Mean Absolute Error
- …

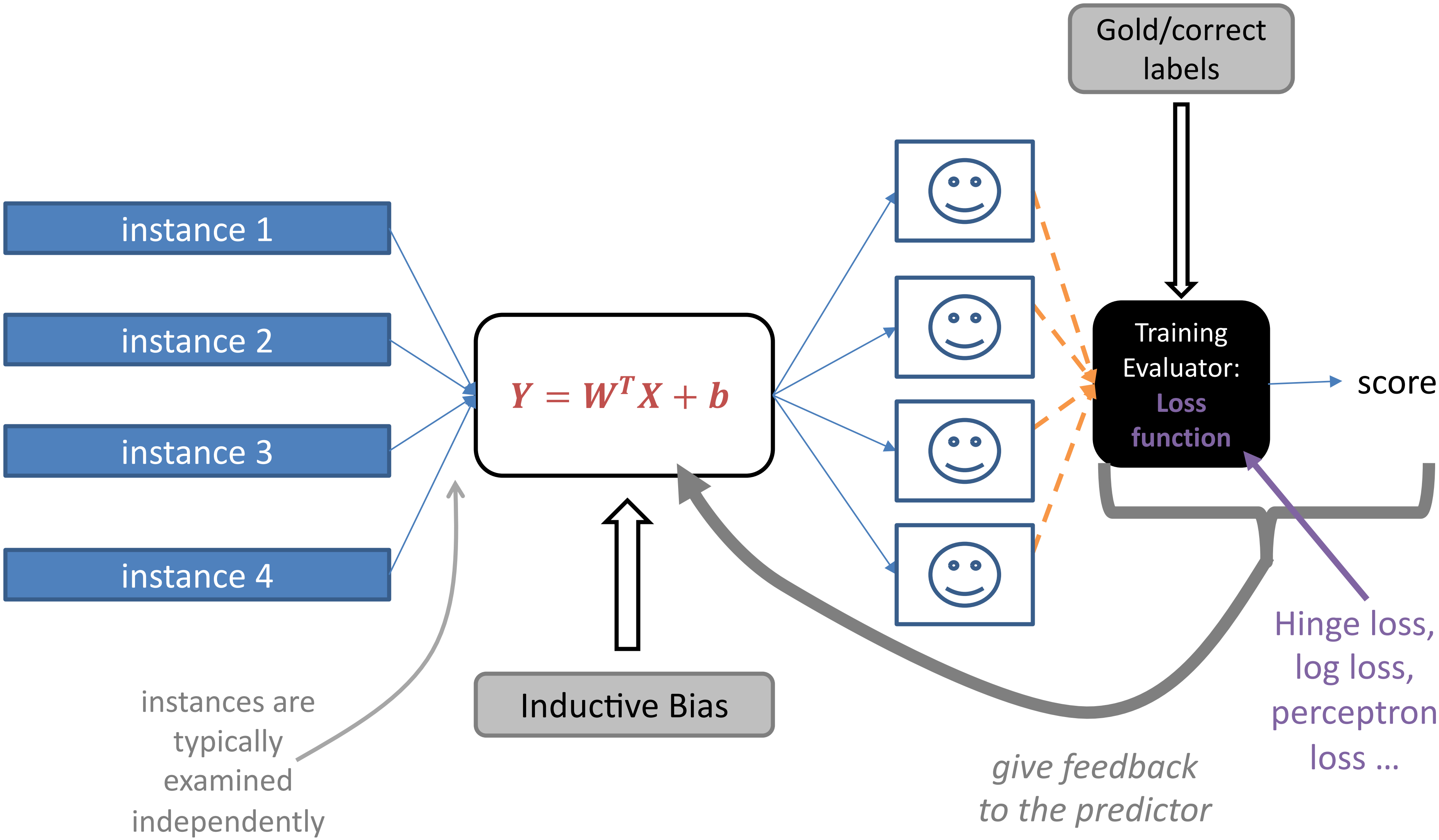
Clustering

- Mutual Information
- V-score
- …

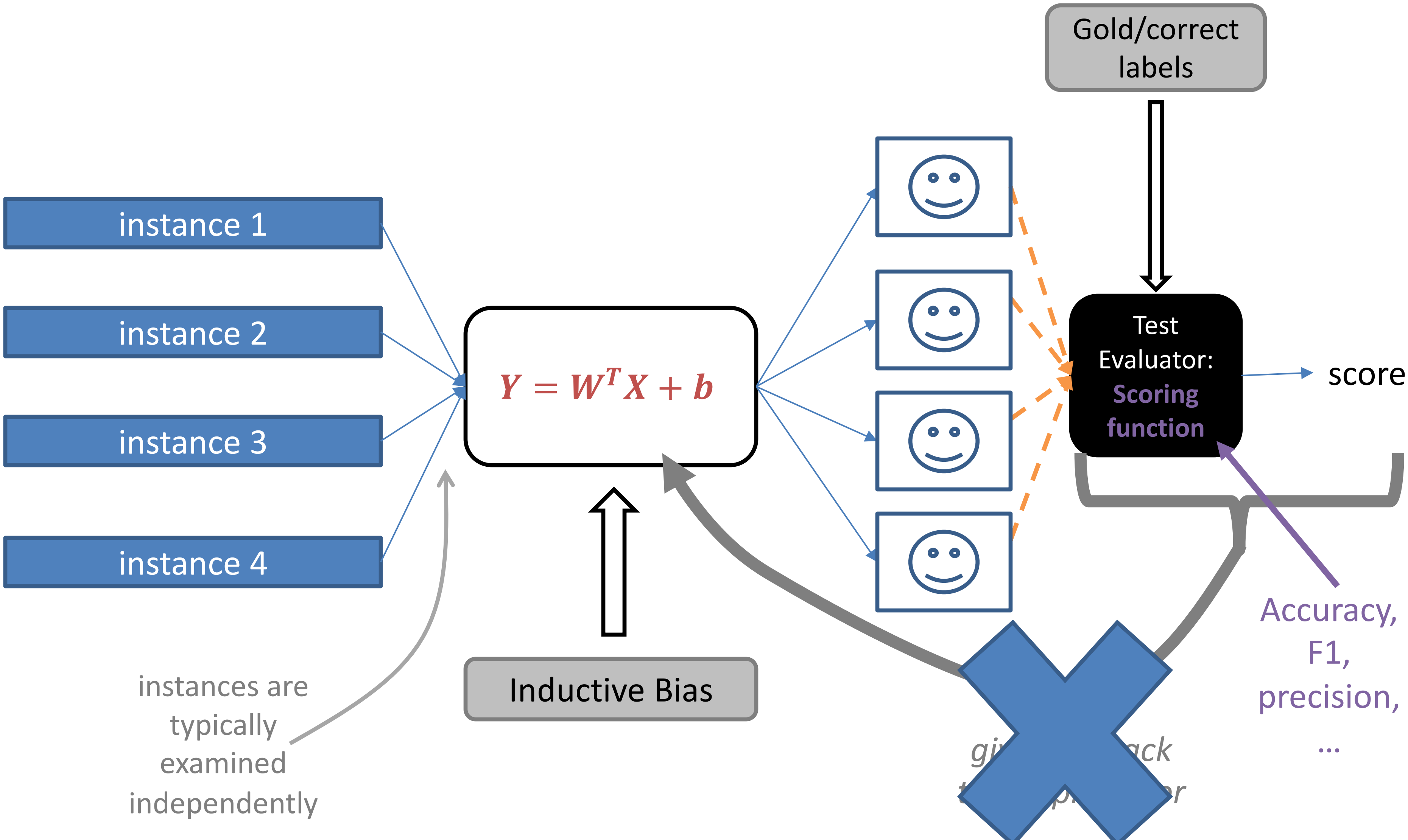
the ***task****: what* kind *of problem are you solving?*

How do we learn these linear classification methods?

Change the loss function. (478/678 topics)



How do we evaluate these linear classification methods? Change the eval function.



What if

- We want a unified way to predict more than two classes?
- We want a probabilistic (bounded, interpretable) score?
- We want to use *transformations* of our data x to help make decisions?

What if

- We want a unified way to predict more than two classes?
- We want a probabilistic (bounded, interpretable) score?
- We want to use *transformations* of our data x to help make decisions?

Terminology

common ML 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

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

Core Aspects to Maxent Classifier p(y|x)

- **features** $f(x, y)$ between x and y that are meaningful;
- **weights** w (one per feature) to say how important each feature is; and
- a way to **form probabilities** from f and w

$$p(y \mid x) = \frac{\exp(\mathrm{w}^T f(x, y))}{\sum_{y'} \exp(\mathrm{w}^T f(x, y'))}$$

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

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

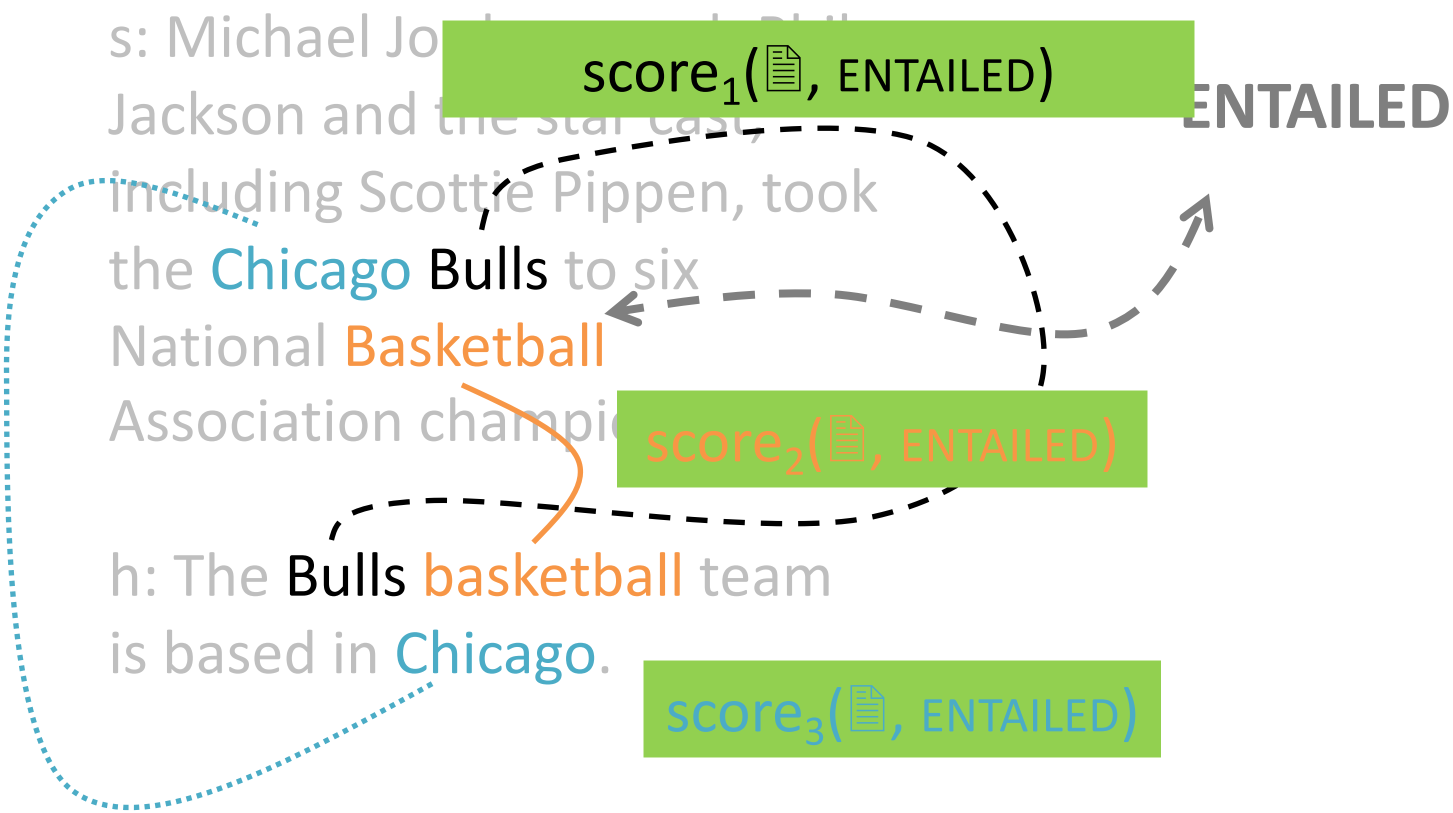
s: Michael Jordan, coach Phil Jackson and the star cast, including Scottie Pippen, took the Chicago Bulls to six National Basketball Association championships.

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

$score_1$(, ENTAILED)

$score_2$(, ENTAILED)

$score_3$(, ENTAILED)

…

$score_k$(, ENTAILED)

…

COMBINE →

posterior probability of ENTAILED

Scoring Our Clues

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) =

(ignore the feature indexing for now)

score_1(, ENTAILED) +

score_2(, ENTAILED) +

score_3(, ENTAILED) +

…

A linear scoring model!

Scoring Our Clues

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) =

Learn these scores… but how?

What do we optimize?

score_1(🗎, ENTAILED) +

score_2(🗎, ENTAILED) +

score_3(🗎, ENTAILED) +

…

A linear scoring model!

Turning Scores into Probabilities (More Generally)

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

$$\Rightarrow$$

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

KEY IDEA

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(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))

A linear scoring model!

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.) $\propto$

exp(
score_1(🗎, ENTAILED) +
score_2(🗎, ENTAILED) +
score_3(🗎, ENTAILED) +
…
))

Maxent Modeling

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

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(score$_1$(🗎, ENTAILED) + score$_2$(🗎, ENTAILED) + score$_3$(🗎, ENTAILED) + …))

Learn the scores (but we'll declare what combinations should be looked at)

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(

$weight_1 * applies_1($, ENTAILED) +

$weight_2 * applies_2($, ENTAILED) +

$weight_3 * applies_3($, ENTAILED) +

…

))

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(
$weight_1 * applies_1$(📄, ENTAILED) +
$weight_2 * applies_2$(📄, ENTAILED) +
$weight_3 * applies_3$(📄, ENTAILED) +
…
))

K different weights… for K different features

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($\text{weight}_1 * \text{applies}_1$(🗎, ENTAILED) +
$\text{weight}_2 * \text{applies}_2$(🗎, ENTAILED) +
$\text{weight}_3 * \text{applies}_3$(🗎, ENTAILED) +
…))

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.

) $\propto$

exp(Dot_product of weight_vec feature_vec(, ENTAILED))

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.

$) \propto$

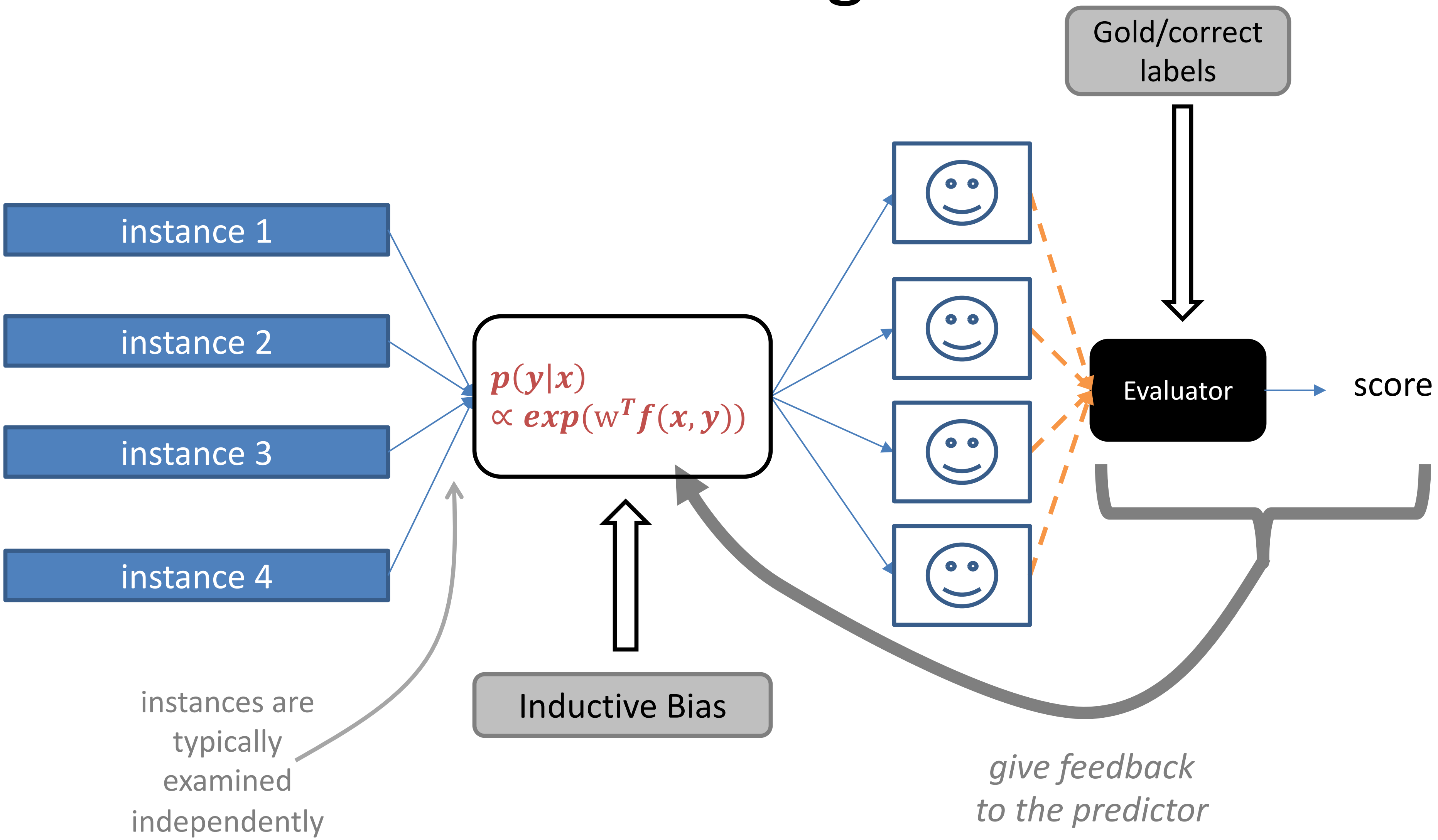
$\exp(\; w^T f($ [document], ENTAILED $) \;)$

K different weights…

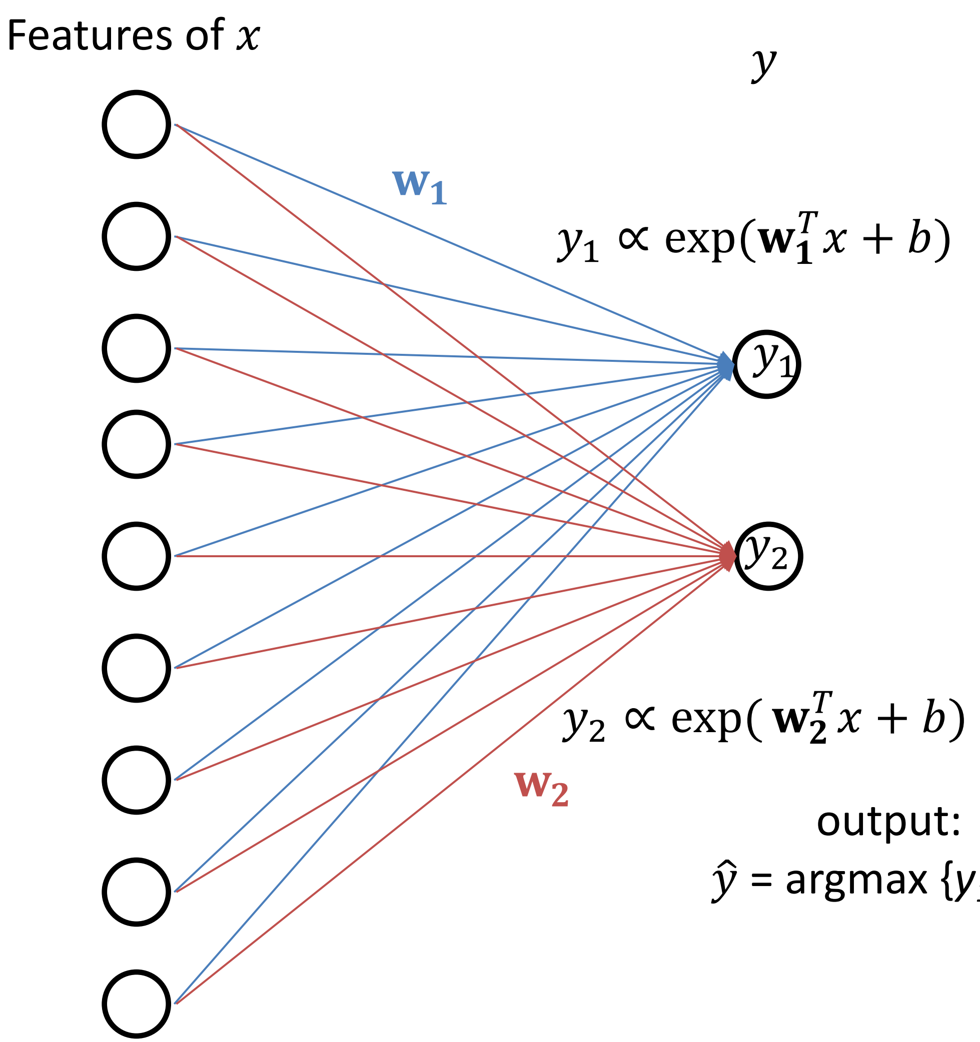
for K different features…

multiplied and then summed

Machine Learning Framework: Learning

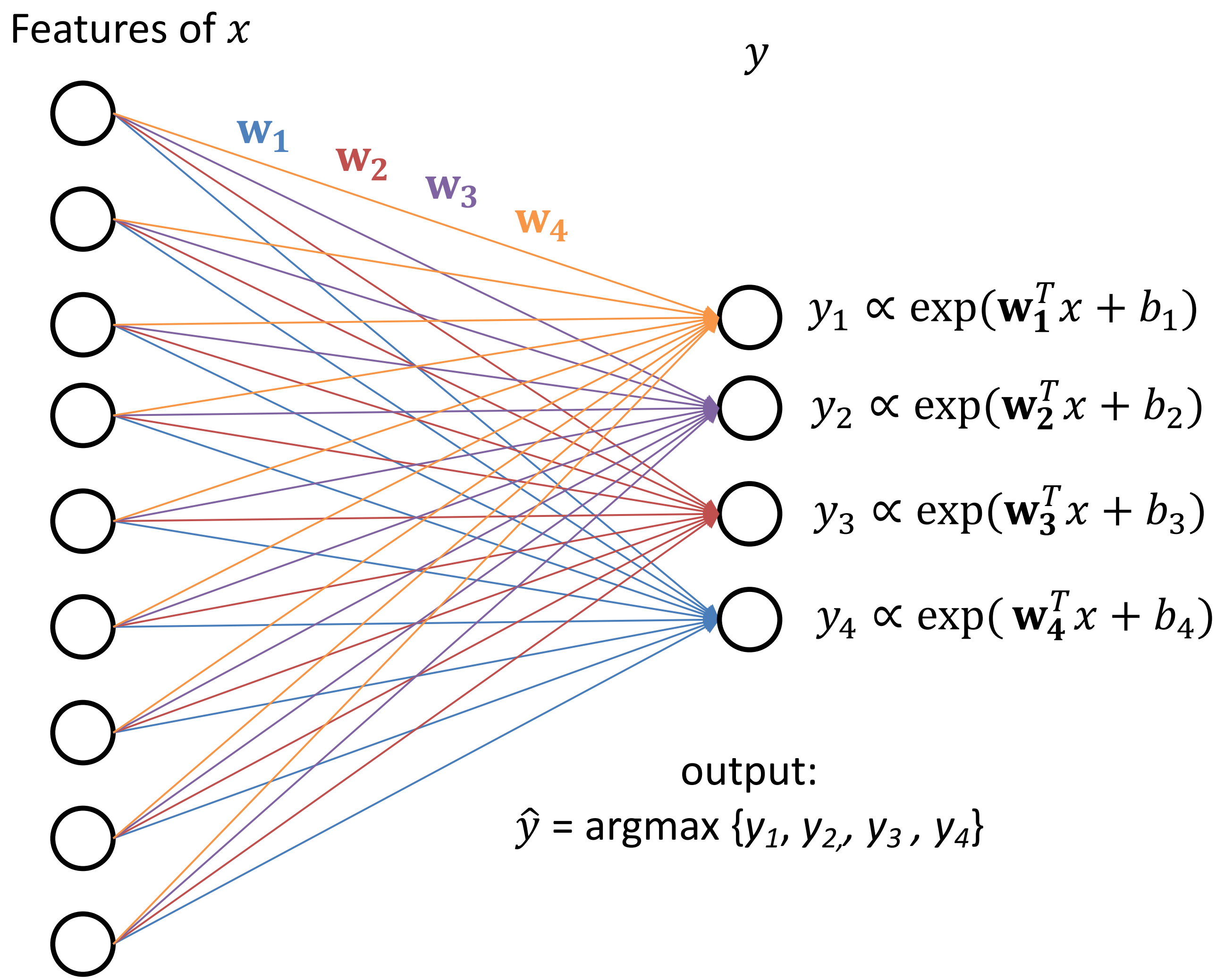


A Graphical View of Logistic Regression/Classification (2 classes)



output:
$\hat{y}$ = argmax $\{y_1, y_2\}$

A Graphical View of Logistic Regression/Classification (4 classes)



output:
$\hat{y}$ = argmax $\{y_1, y_2, y_3, y_4\}$

sklearn.linear_model.LogisticRegression ¶

class `sklearn.linear_model.LogisticRegression`(*penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Read more in the User Guide.

Parameters:

penalty : *{'l1', 'l2', 'elasticnet', 'none'}, default='l2'*

Used to specify the norm used in the penalization. The 'newton-cg', 'sag' and 'lbfgs' solvers support only l2 penalties. 'elasticnet' is only supported by the 'saga' solver. If 'none' (not supported by the liblinear solver), no regularization is applied.

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html