

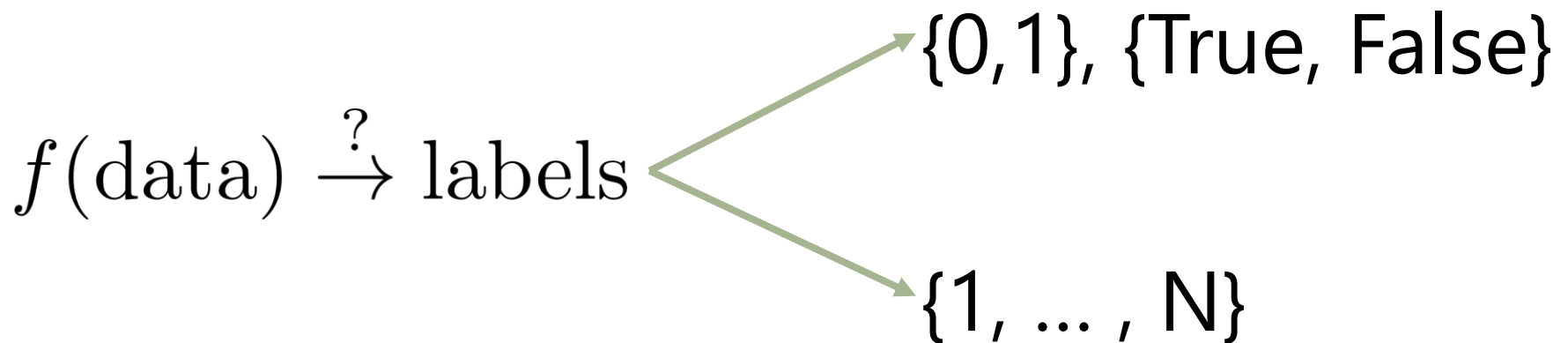
CSE 158 – Lecture 4

Web Mining and Recommender Systems

More Classifiers

Last lecture...

How can we predict **binary** or **categorical** variables?











Last lecture...



Will I **purchase**
this product?
(yes)

Shop for engagement rings on Google Sponsored ⓘ

 <p>French-Set Halo Diamond... \$1,990.00 Ritani</p>	 <p>18K White Gold Delicate... \$950.00 Brilliant Earth ★★★★★ (57)</p>	 <p>18K White Gold Fancy D... \$1,825.00 Brilliant Earth ★★★★★ (13)</p>	 <p>Chamise Diamond Eng... \$975.00 Brilliant Earth ★★★★★ (7)</p>
 <p>Vintage Cushion Halo... \$4,140.00</p>	 <p>Princess Cut Diamond Eng... \$1,906.82</p>	 <p>18K White Gold Hudson... \$975.00</p>	 <p>18K White Gold Harmon... \$1,675.00</p>

Will I **click on**
this ad?
(no)

Last lecture...

- **Naïve Bayes**

- Probabilistic model (fits $p(\text{label}|\text{data})$)
- Makes a conditional independence assumption of the form $(\text{feature}_i \perp\!\!\!\perp \text{feature}_j | \text{label})$ allowing us to define the model by computing $p(\text{feature}_i | \text{label})$ for each feature
- Simple to compute just by counting

- **Logistic Regression**

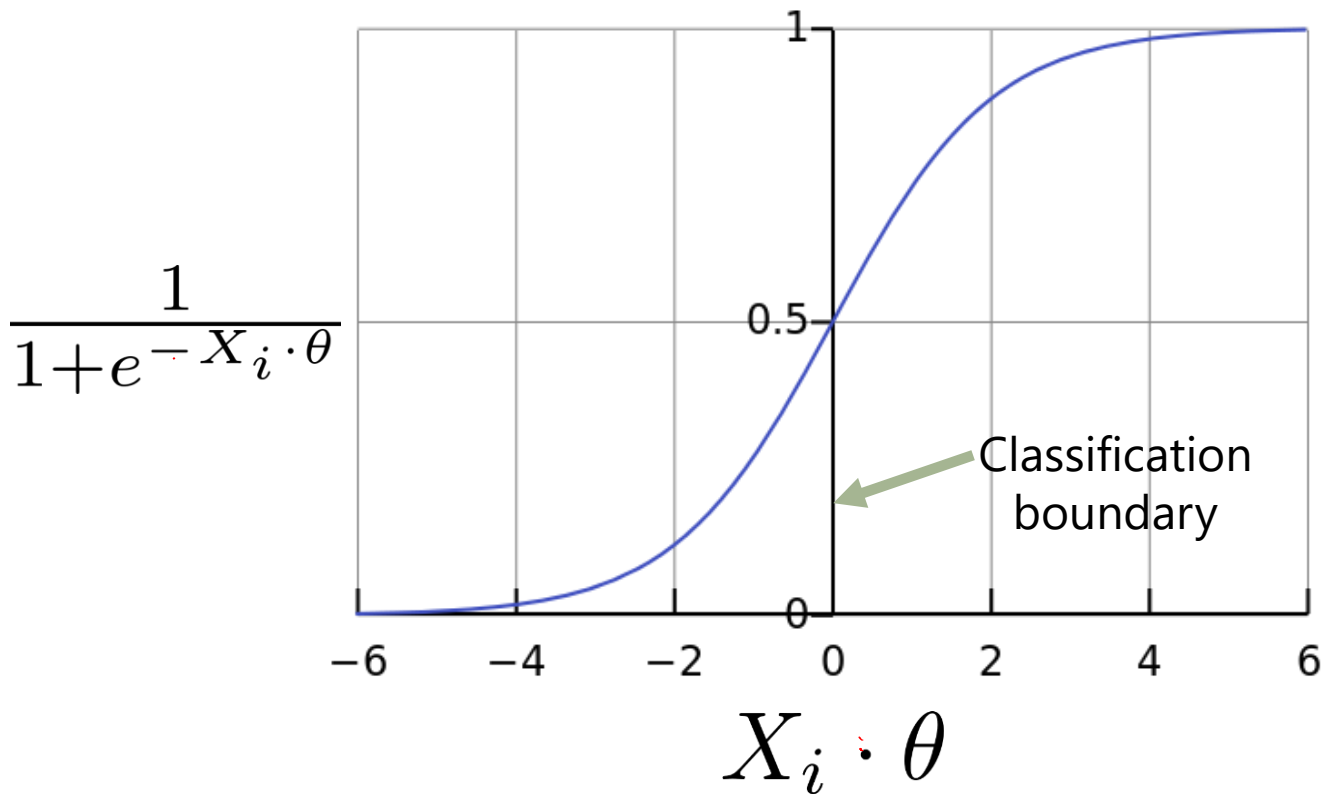
- Fixes the “double counting” problem present in naïve Bayes

- **SVMs**

- Non-probabilistic: optimizes the classification error rather than the likelihood

2) logistic regression

sigmoid function: $\sigma(t) = \frac{1}{1+e^{-t}}$

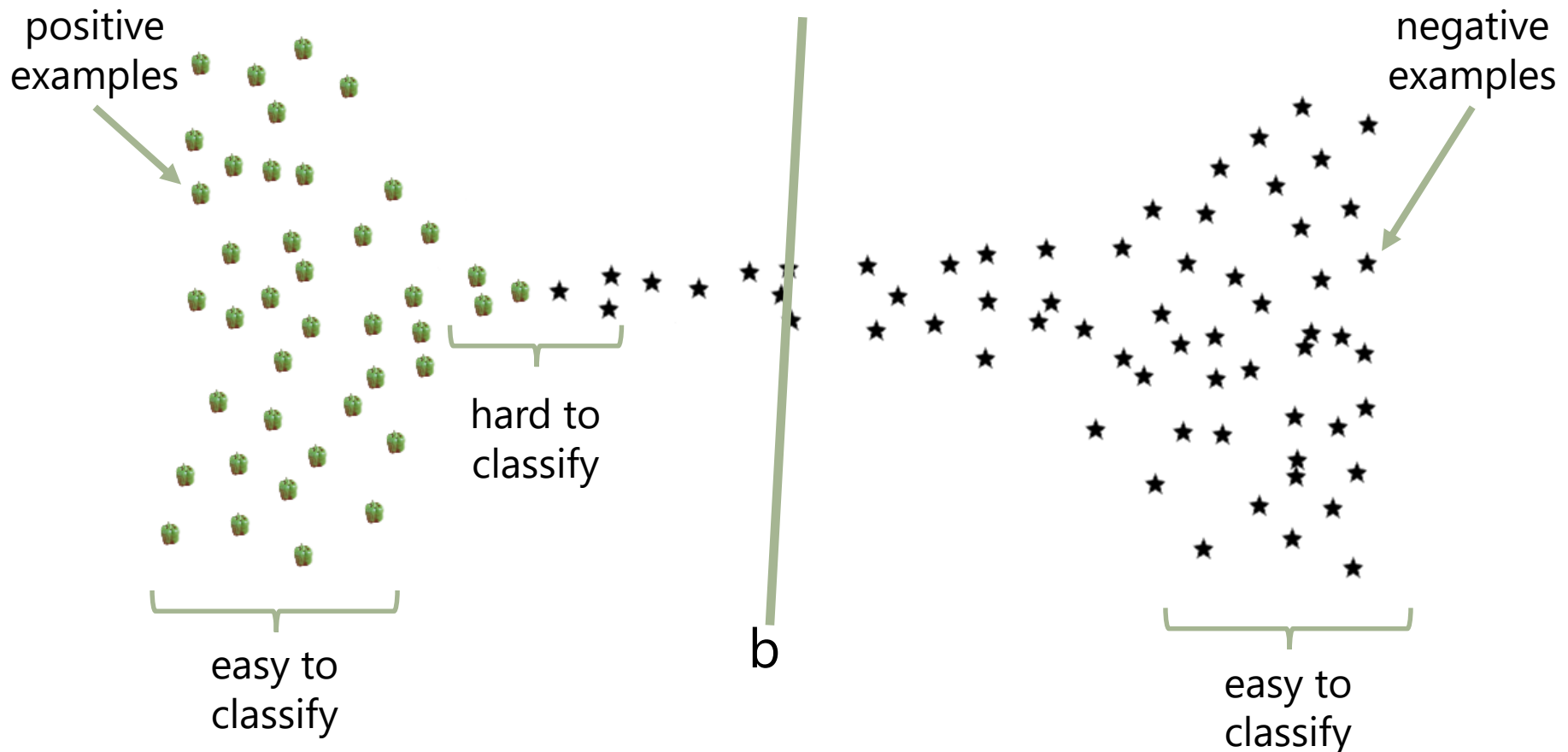


Logistic regression

- Logistic regressors don't optimize the number of "mistakes"
- No special attention is paid to the "difficult" instances – every instance influences the model
- But "easy" instances can affect the model (and in a bad way!)
- How can we develop a classifier that optimizes the number of mislabeled examples?

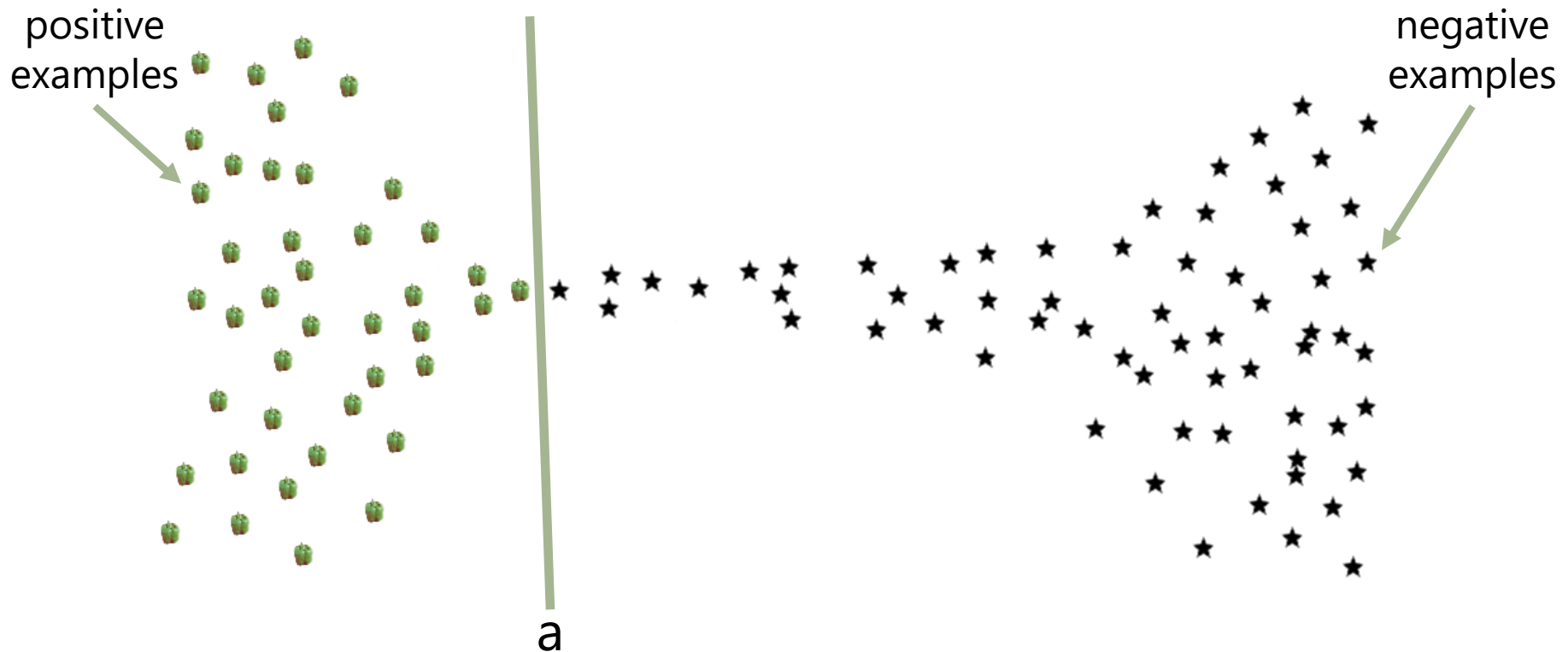
Logistic regression

Q: Where would a logistic regressor place the decision boundary for these features?



3) Support Vector Machines

Try to optimize the **misclassification error** rather than maximize a probability



Support Vector Machines

This is essentially the intuition behind Support Vector Machines (SVMs) – train a classifier that focuses on the “difficult” examples by minimizing the misclassification error

We still want a classifier of the form

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta - \alpha > 0 \\ -1 & \text{otherwise} \end{cases}$$

Handwritten notes: θ_0 points to α ; α and -1 are circled in red.

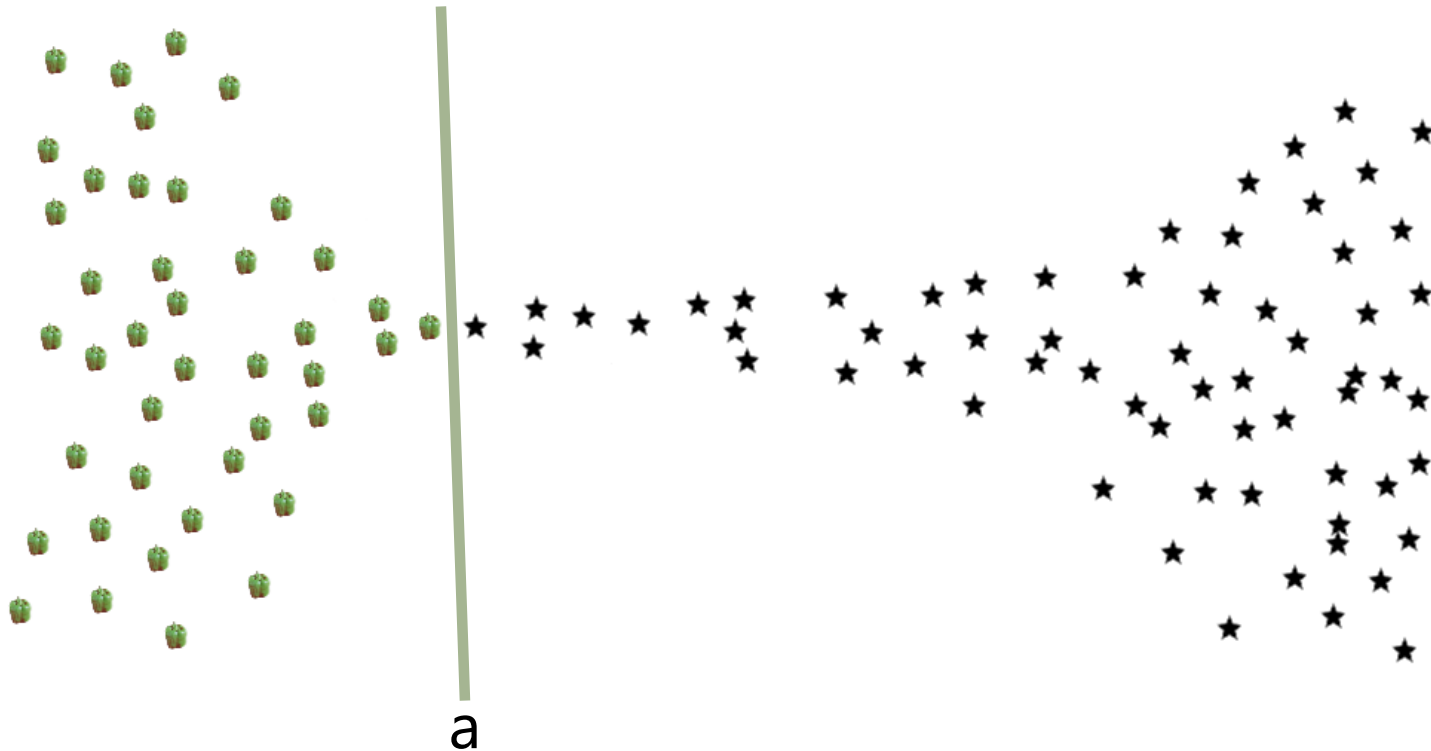
But we want to minimize the number of misclassifications:

$$\arg \min_{\theta} \sum_i \delta(y_i (X_i \cdot \theta - \alpha) \leq 0)$$

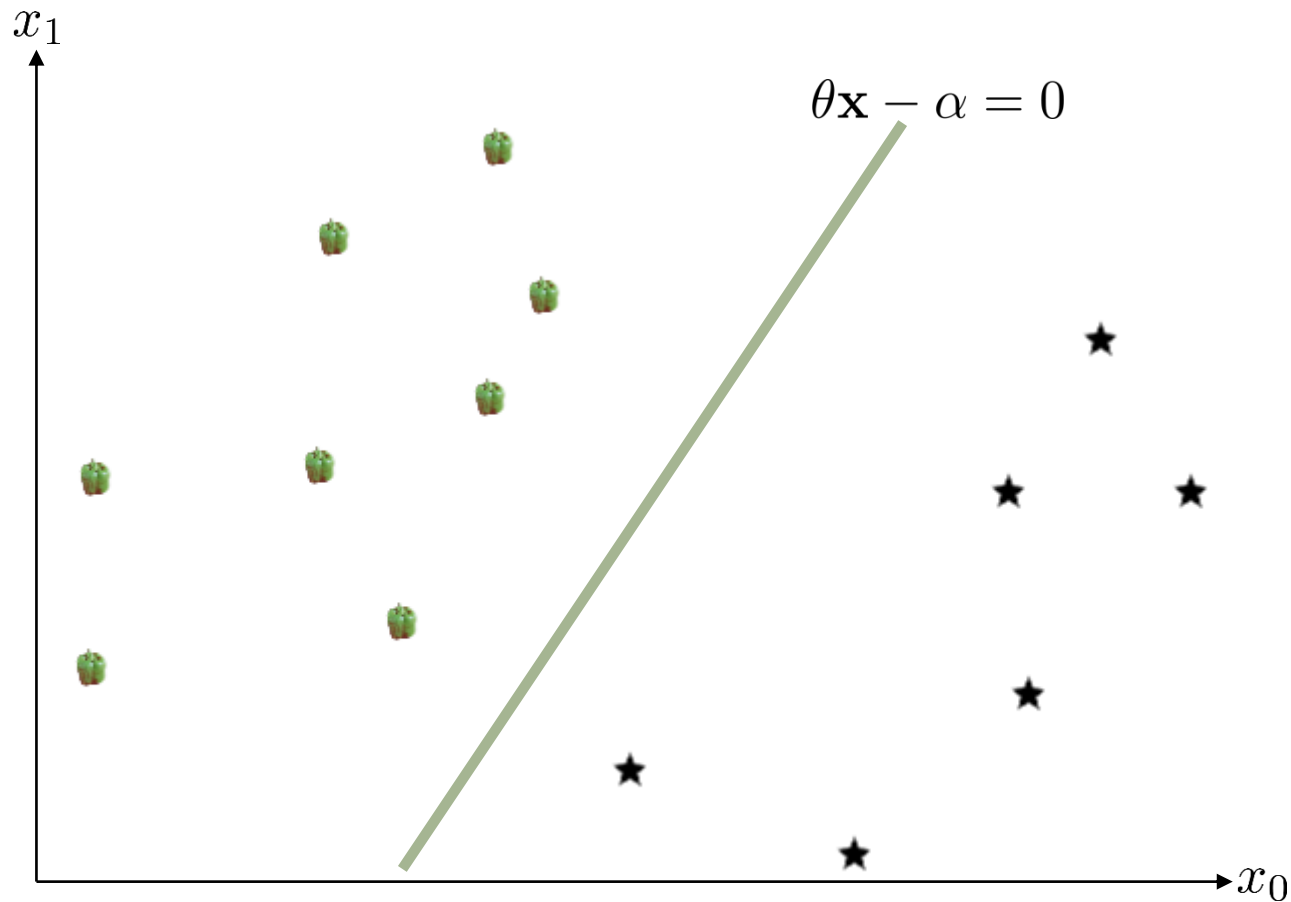
Handwritten note: -1 iff argument is true else 0

Support Vector Machines

Simple (seperable) case: there exists a perfect classifier

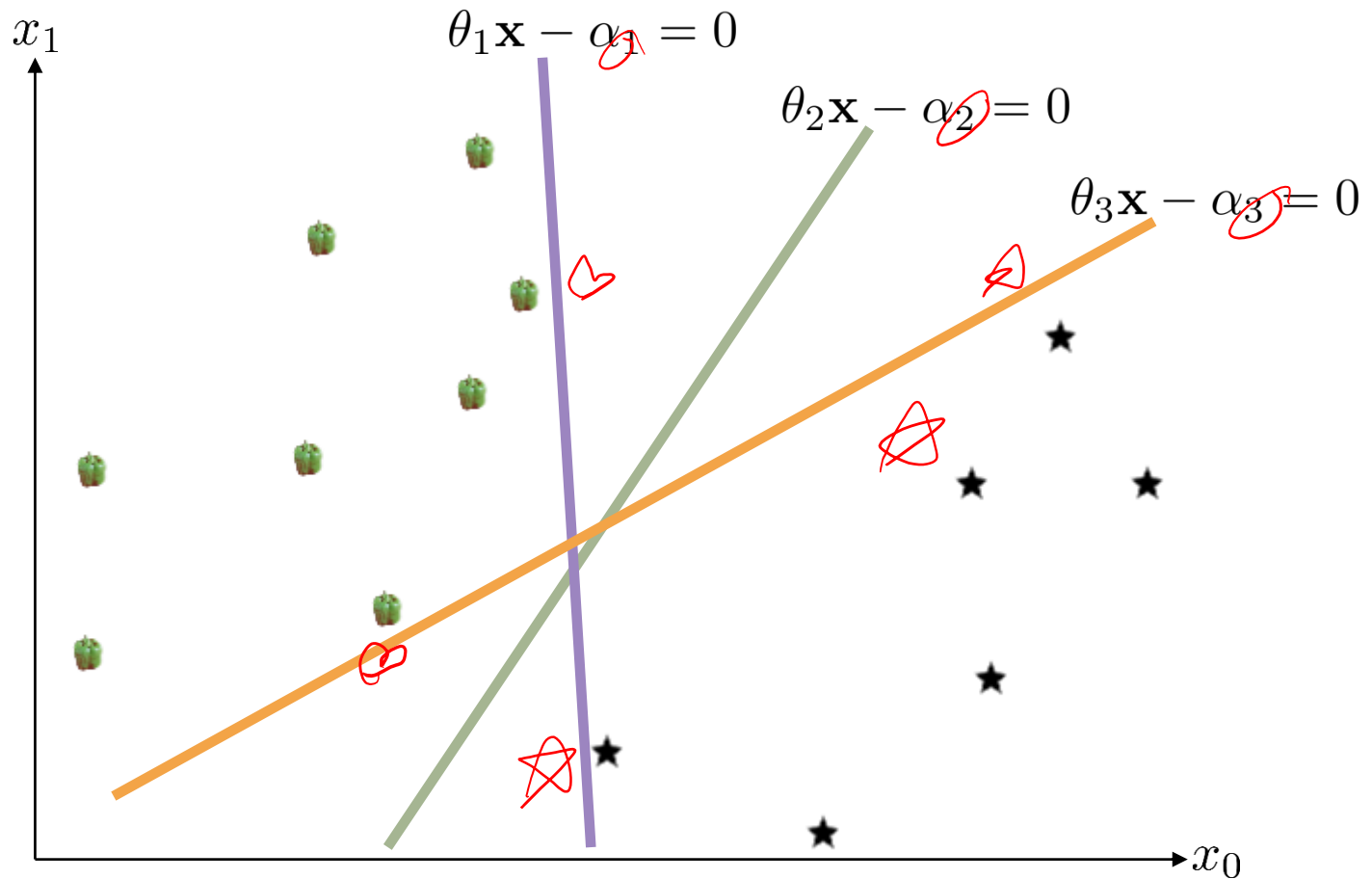


Support Vector Machines



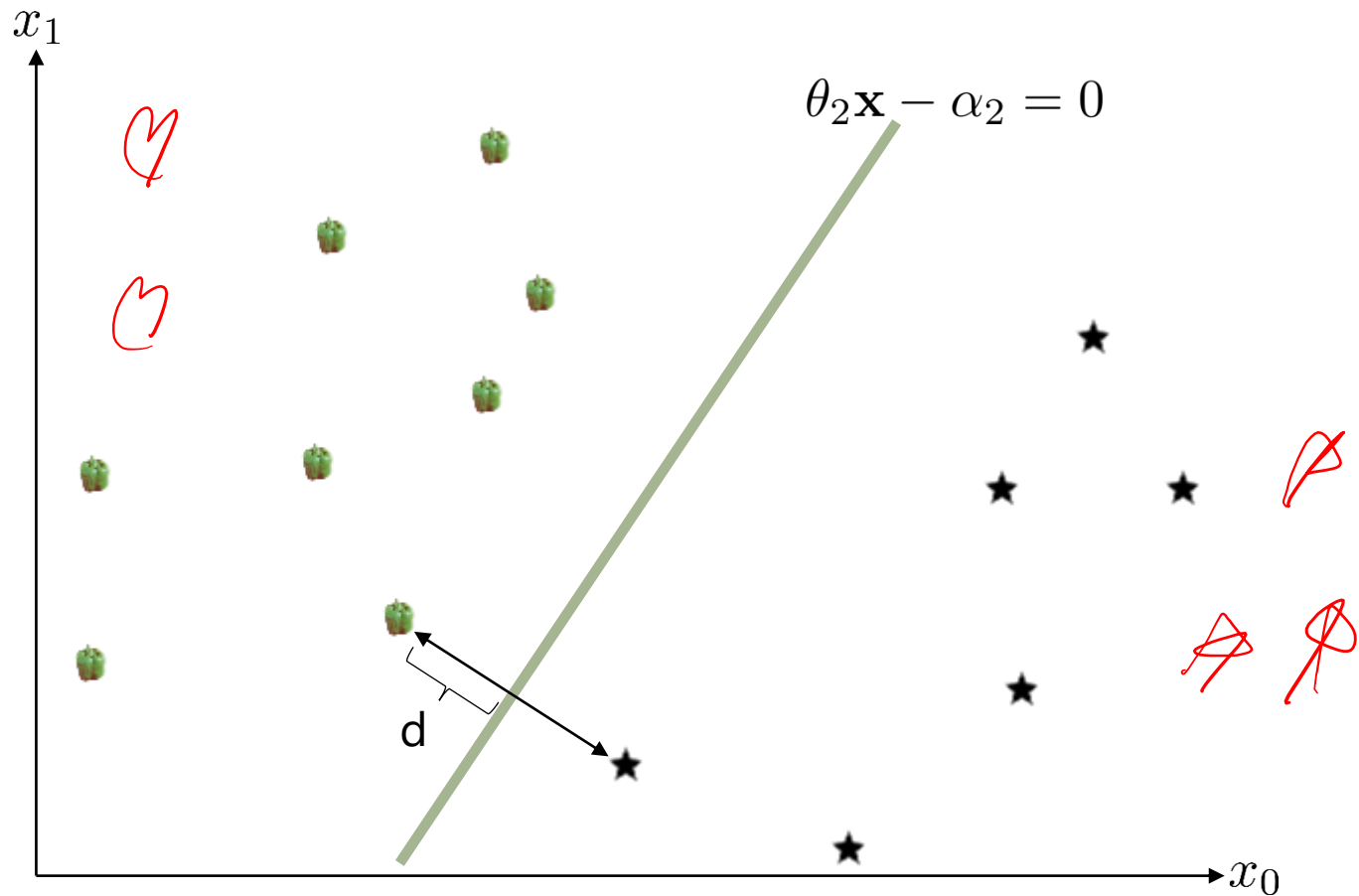
The classifier is defined by the hyperplane $\theta \mathbf{x} - \alpha = 0$

Support Vector Machines



Q: Is one of these classifiers preferable over the others?

Support Vector Machines



A: Choose the classifier that maximizes the distance to the nearest point

Support Vector Machines

Distance from a point to a line?

$$l_n: ax + by + c = 0$$

$$pt: (x_0, y_0)$$

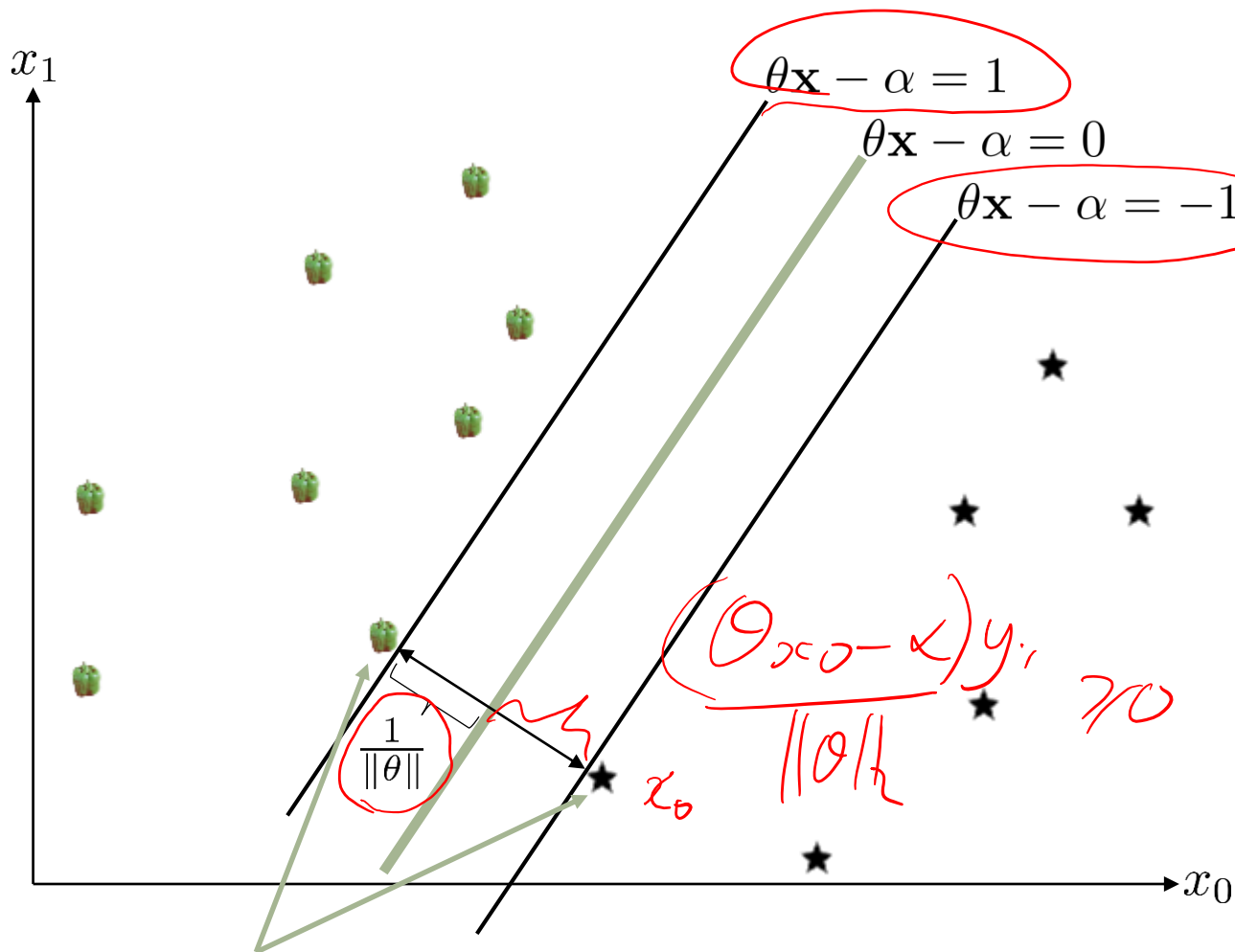
$$d(l_n, pt) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

$$l_n: 0x - x = 0$$

$$pt: \underline{x_0}$$

$$d(l_n, pt) = \frac{|0x_0 - x_0|}{\|0\|_2}$$

Support Vector Machines



"support vectors"

$$\arg \min_{\theta, \alpha} \frac{1}{2} \|\theta\|_2^2$$

such that

$$\forall_i y_i (\theta \cdot X_i - \alpha) \geq 1$$

Support Vector Machines

This is known as a
"quadratic program" (QP)
and can be solved using
"standard" techniques

$$\arg \min_{\theta, \alpha} \frac{1}{2} \|\theta\|_2^2$$

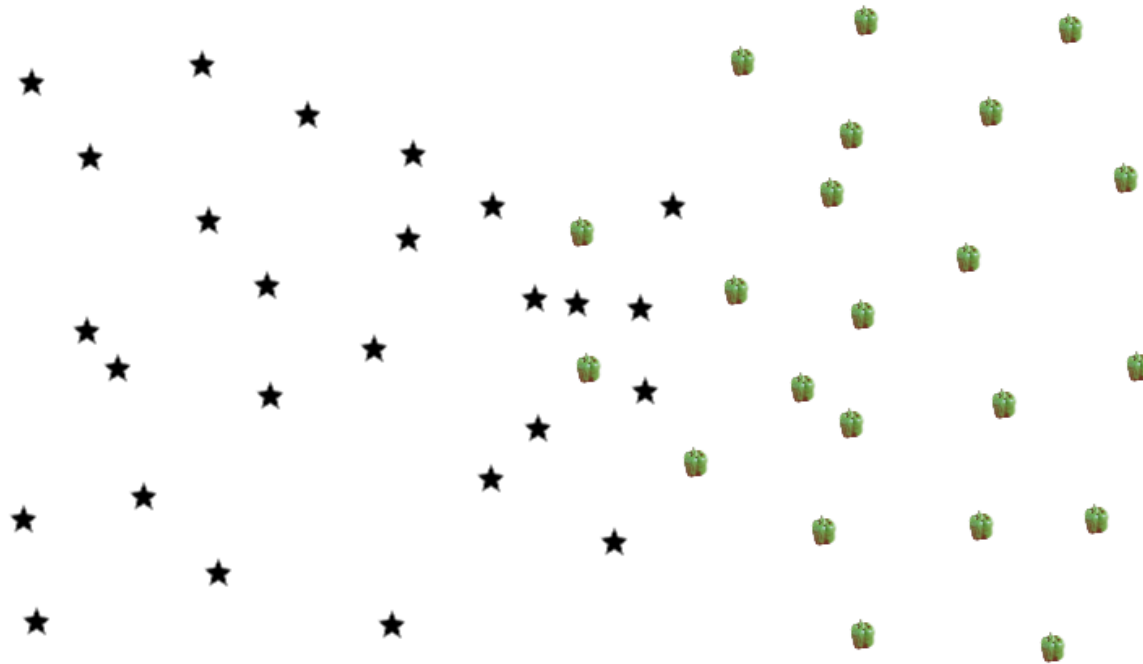
such that

$$\forall_i y_i (\theta \cdot X_i - \alpha) \geq 1$$

See e.g. Nocedal & Wright ("Numerical Optimization"), 2006

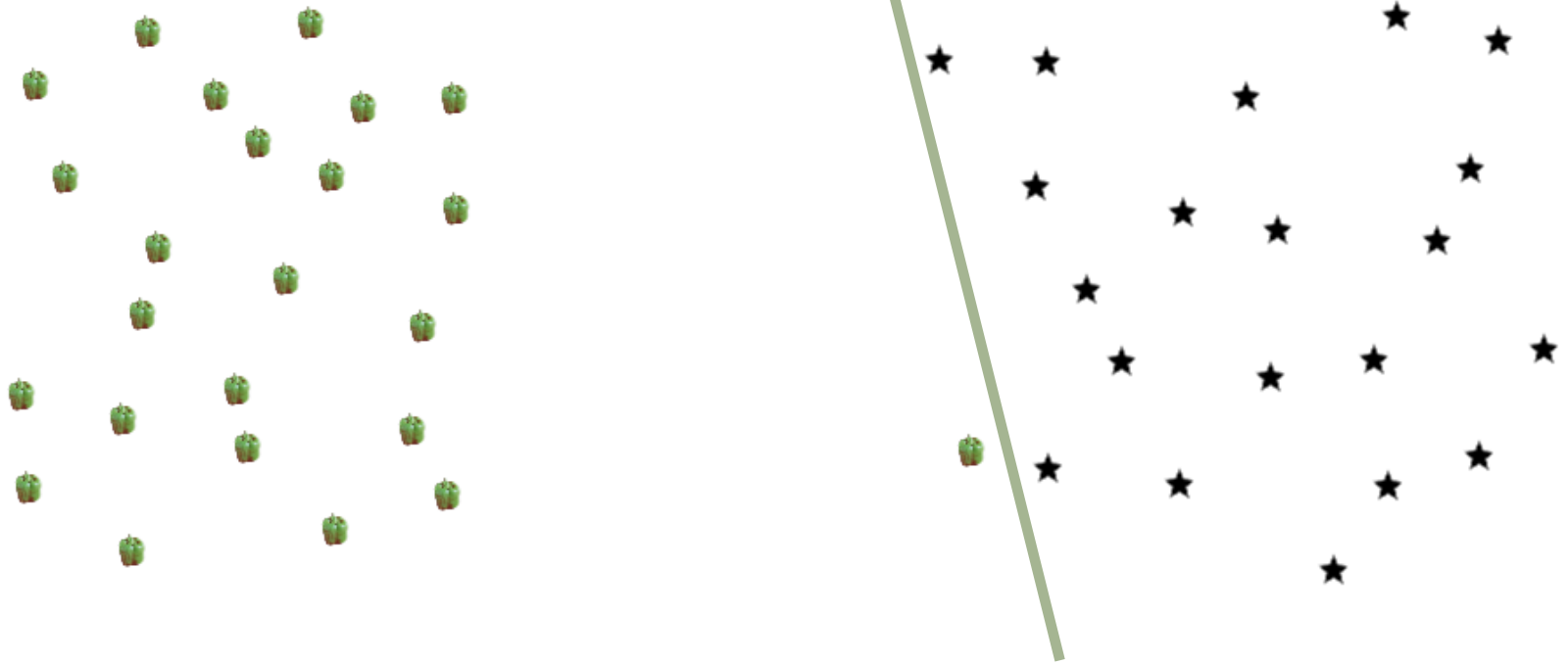
Support Vector Machines

But: is finding such a separating hyperplane even possible?



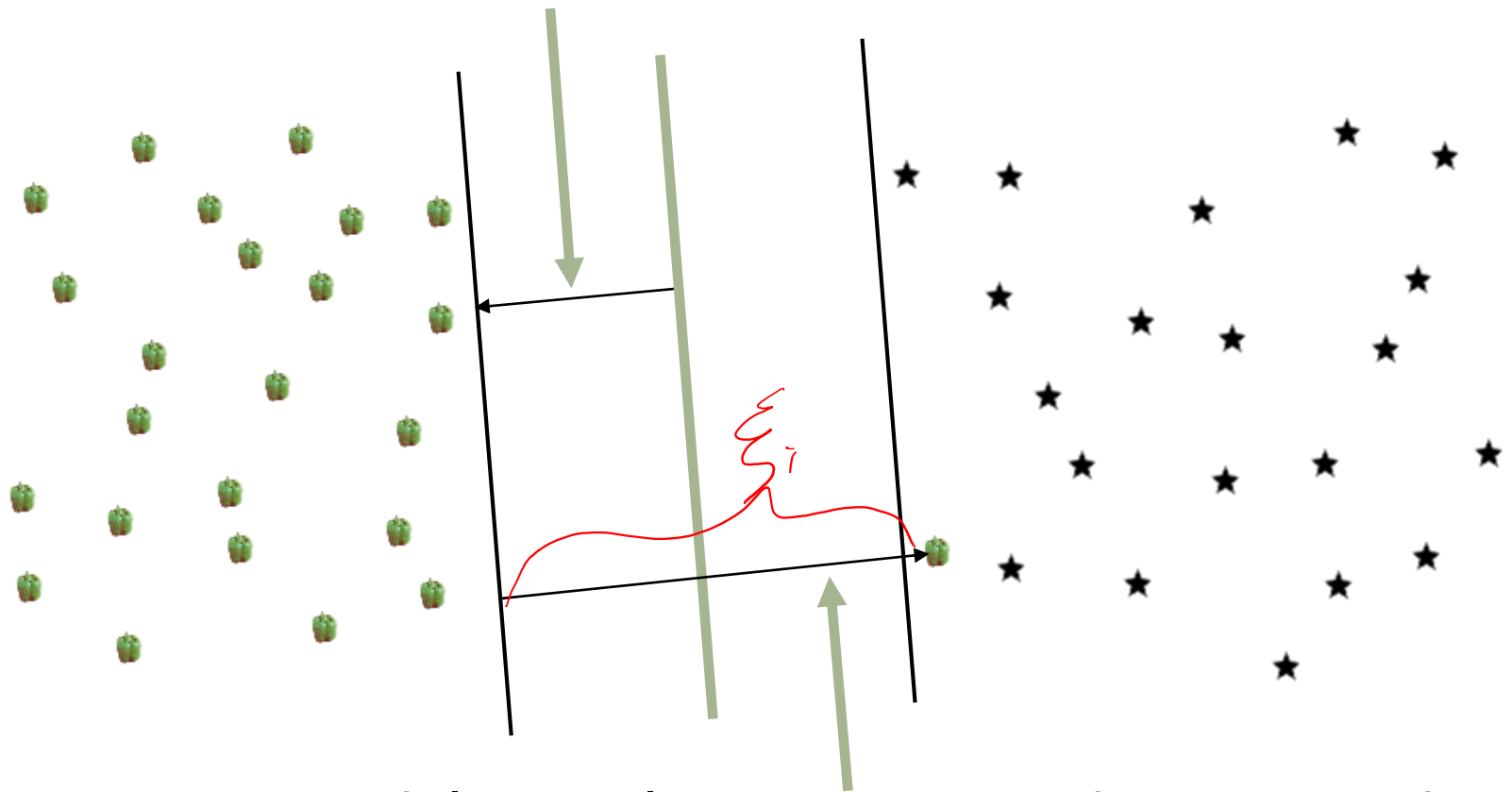
Support Vector Machines

Or: is it actually a good idea?



Support Vector Machines

Want the margin to be as wide as possible



While penalizing points on the wrong side of it

Support Vector Machines

Soft-margin formulation:

$$\arg \min_{\theta, \alpha, \xi_i} \frac{1}{2} \|\theta\|_2^2 + \sum_i \xi_i$$

such that

$$\forall_i y_i (\theta \cdot X_i - \alpha) \geq 1 - \xi_i$$

Summary

The classifiers we've seen this week all attempt to make decisions by associating weights (θ) with features (x) and classifying according to

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

Summary

- **Naïve Bayes**

- Probabilistic model (fits $p(\text{label}|\text{data})$)
- Makes a conditional independence assumption of the form $(\text{feature}_i \perp\!\!\!\perp \text{feature}_j | \text{label})$ allowing us to define the model by computing $p(\text{feature}_i | \text{label})$ for each feature
- Simple to compute just by counting

- **Logistic Regression**

- Fixes the “double counting” problem present in naïve Bayes

- **SVMs**

- Non-probabilistic: optimizes the classification error rather than the likelihood

Pros/cons

- **Naïve Bayes**

- ++ Easiest to implement, most efficient to “train”
- ++ If we have a process that generates feature that *are* independent given the label, it’s a very sensible idea
- Otherwise it suffers from a “double-counting” issue

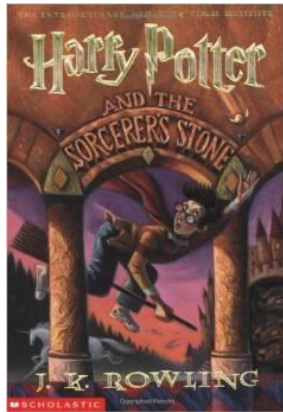
- **Logistic Regression**

- ++ Fixes the “double counting” problem present in naïve Bayes
- More expensive to train

- **SVMs**

- ++ Non-probabilistic: optimizes the classification error rather than the likelihood
- More expensive to train

Judging a book by its cover



[0.723845, 0.153926, 0.757238, 0.983643, ...]

4096-dimensional image features

Images features are available for each book on
http://jmcauley.ucsd.edu/cse158/data/amazon/book_images_5000.json



<http://caffe.berkeleyvision.org/>

Judging a book by its cover

Example: train an SVM to predict whether a book is a children's book from its cover art

(code available on)

<http://jmcauley.ucsd.edu/cse158/code/week2.py>

Judging a book by its cover

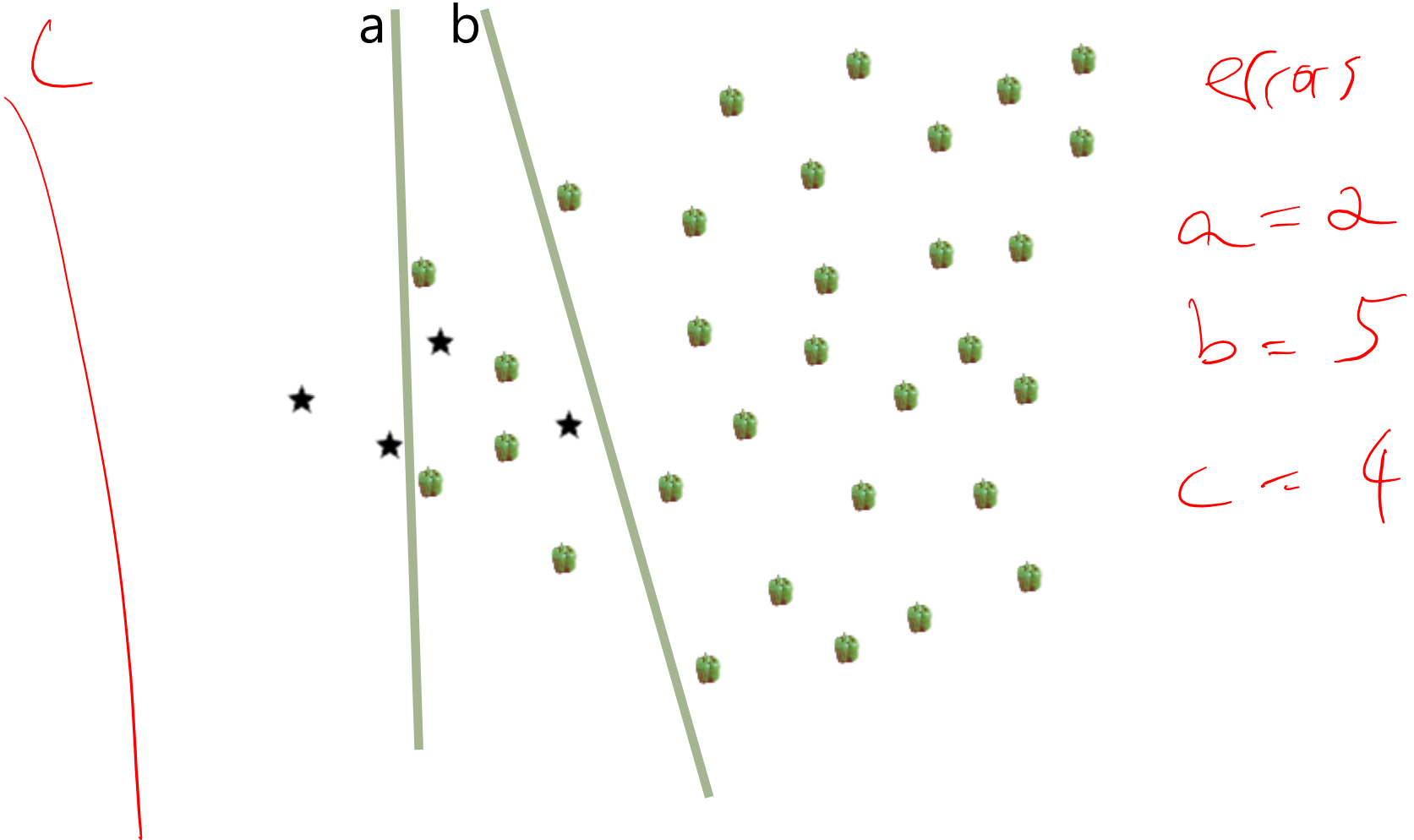
- The number of errors we made was extremely low, yet our classifier doesn't seem to be very good – why?

CSE 158 – Lecture 4

Web Mining and Recommender Systems

Evaluating Classifiers

Which of these classifiers is best?



Which of these classifiers is best?

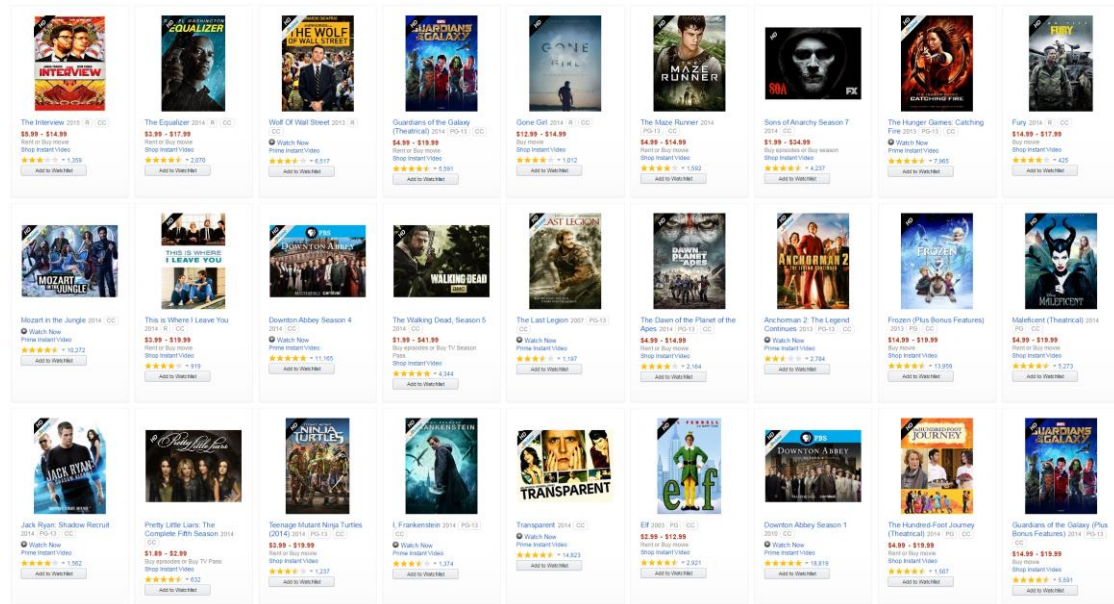
The solution which minimizes the #errors may not be the best one

Which of these classifiers is best?

1. When data are highly imbalanced

If there are far fewer positive examples than negative examples we may want to assign additional weight to negative instances (or vice versa)

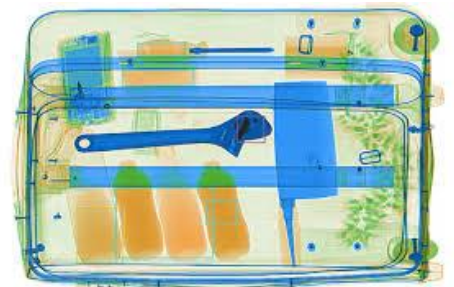
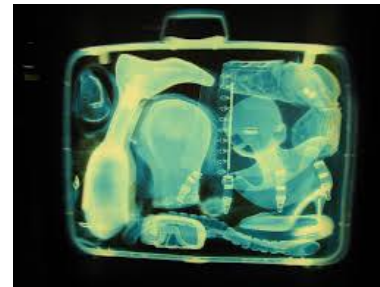
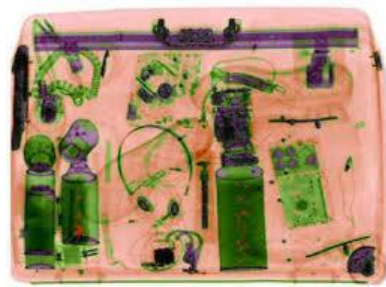
e.g. will I purchase a product? If I purchase 0.00001% of products, then a classifier which just predicts "no" everywhere is 99.99999% accurate, but not very useful



Which of these classifiers is best?

2. When mistakes are more costly in one direction

False positives are nuisances but false negatives are disastrous (or vice versa)

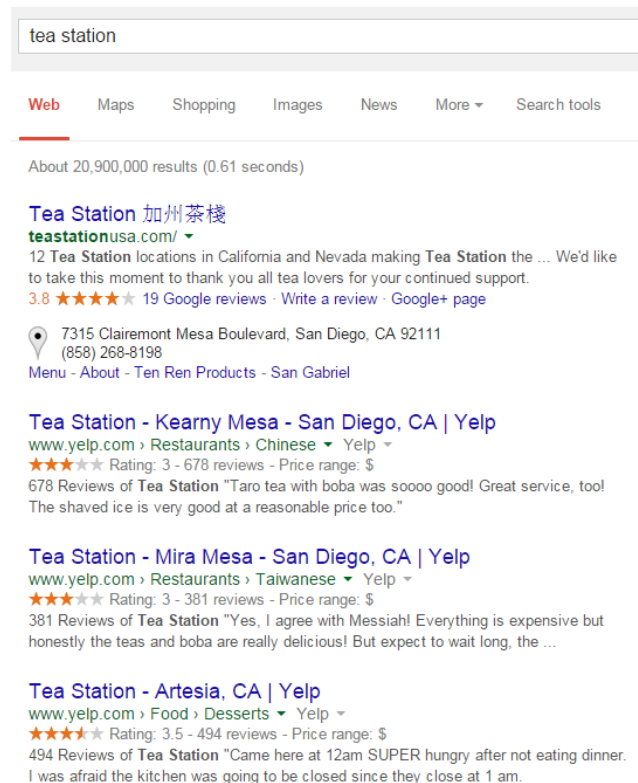


e.g. which of these bags contains a weapon?

Which of these classifiers is best?

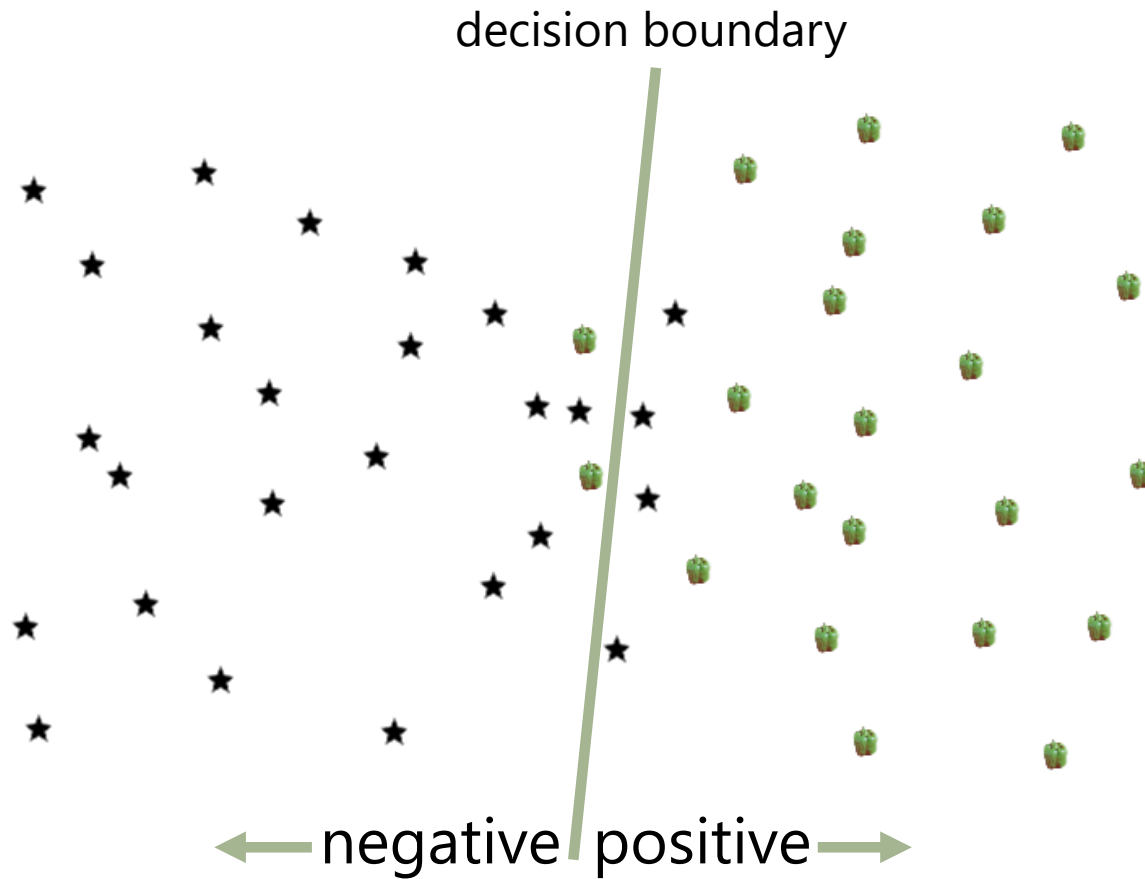
3. When we only care about the “most confident” predictions

e.g. does a relevant result appear among the first page of results?

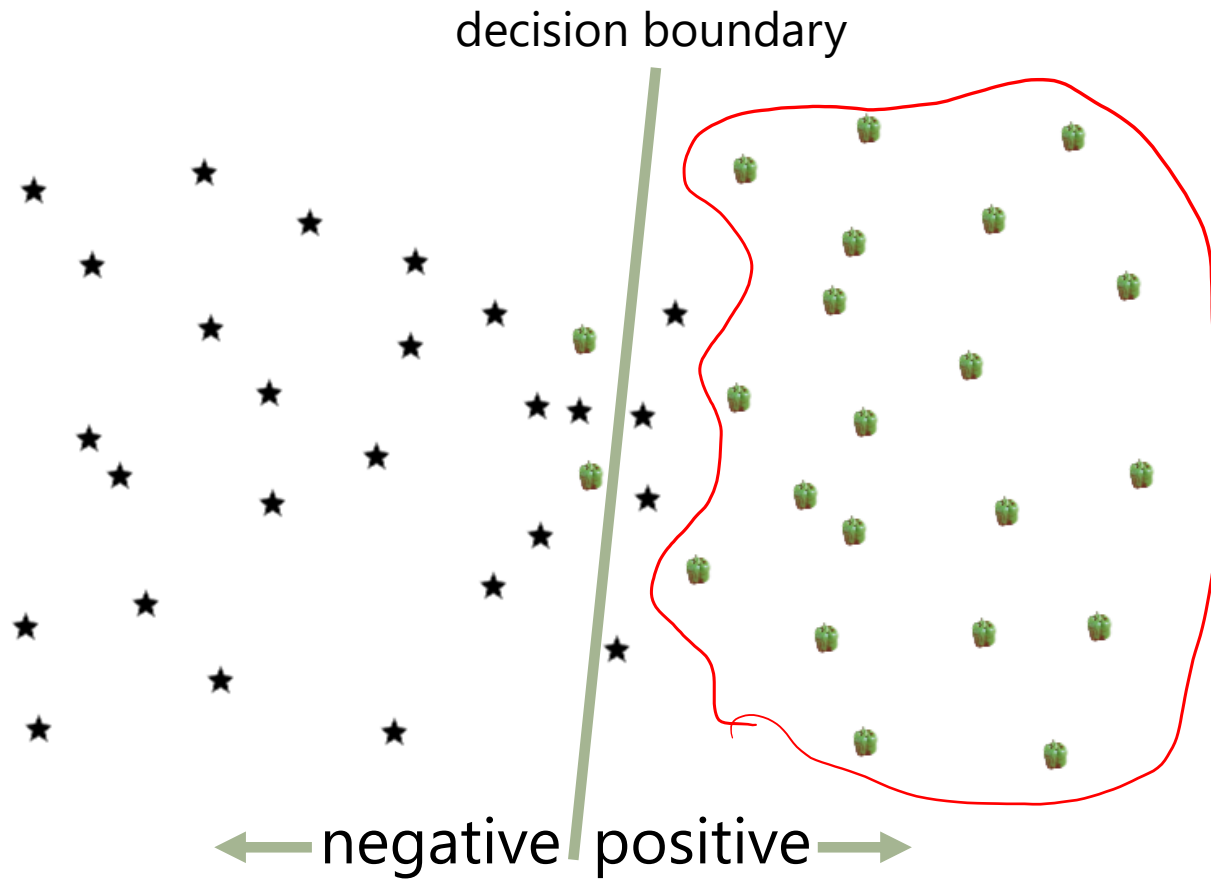


The screenshot shows a search engine results page for the query "tea station". The search bar at the top contains the text "tea station". Below the search bar, there are navigation tabs for "Web", "Maps", "Shopping", "Images", "News", "More", and "Search tools". The search results are displayed below, showing approximately 20,900,000 results in 0.61 seconds. The first result is for "Tea Station 加州茶棧" (Tea Station California Tea House) with the URL "teastationusa.com/". The description mentions 12 locations in California and Nevada and includes a 3.8-star rating from 19 Google reviews. The address is 7315 Clairemont Mesa Boulevard, San Diego, CA 92111, with a phone number (858) 268-8198. The second result is for "Tea Station - Kearny Mesa - San Diego, CA | Yelp" with a 3-star rating and 678 reviews. The third result is for "Tea Station - Mira Mesa - San Diego, CA | Yelp" with a 3-star rating and 381 reviews. The fourth result is for "Tea Station - Artesia, CA | Yelp" with a 3.5-star rating and 494 reviews.

Evaluating classifiers

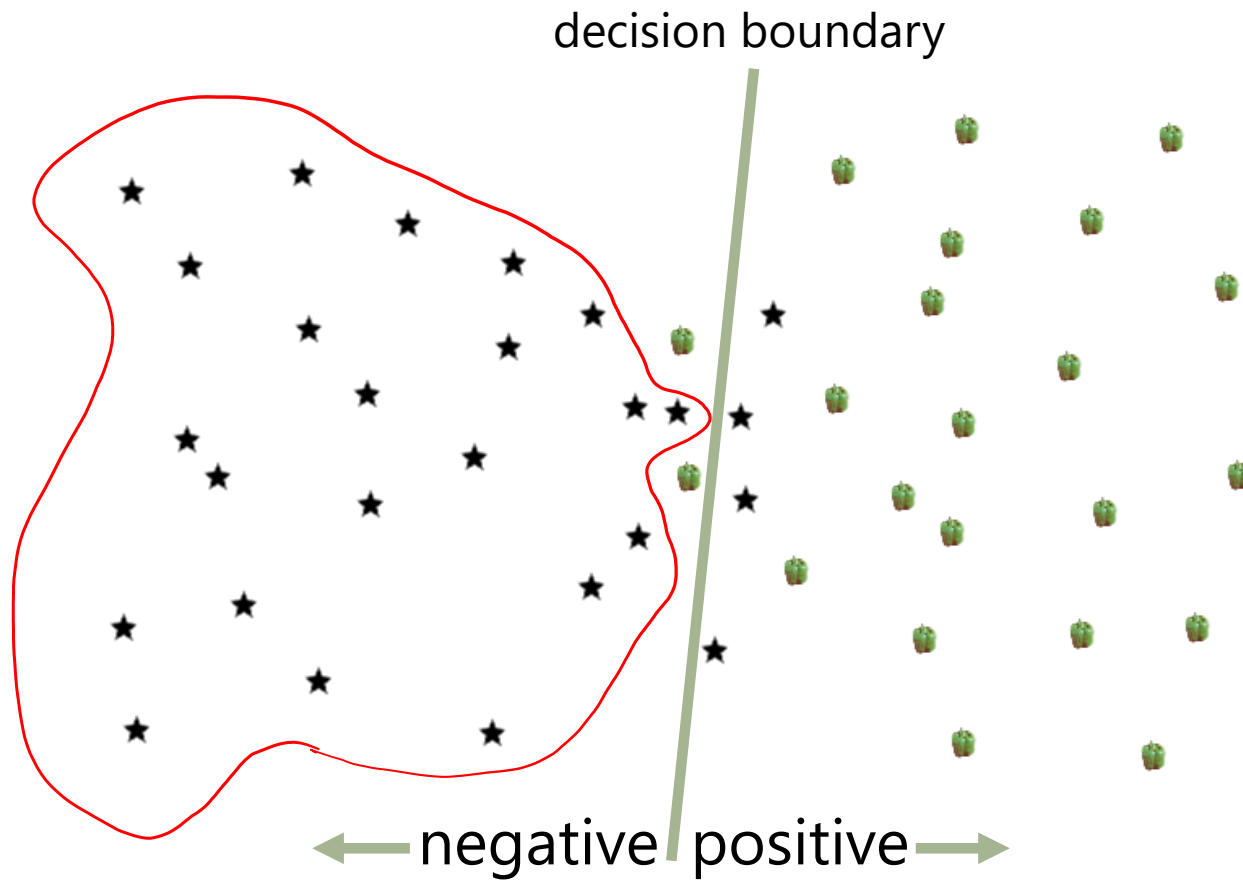


Evaluating classifiers



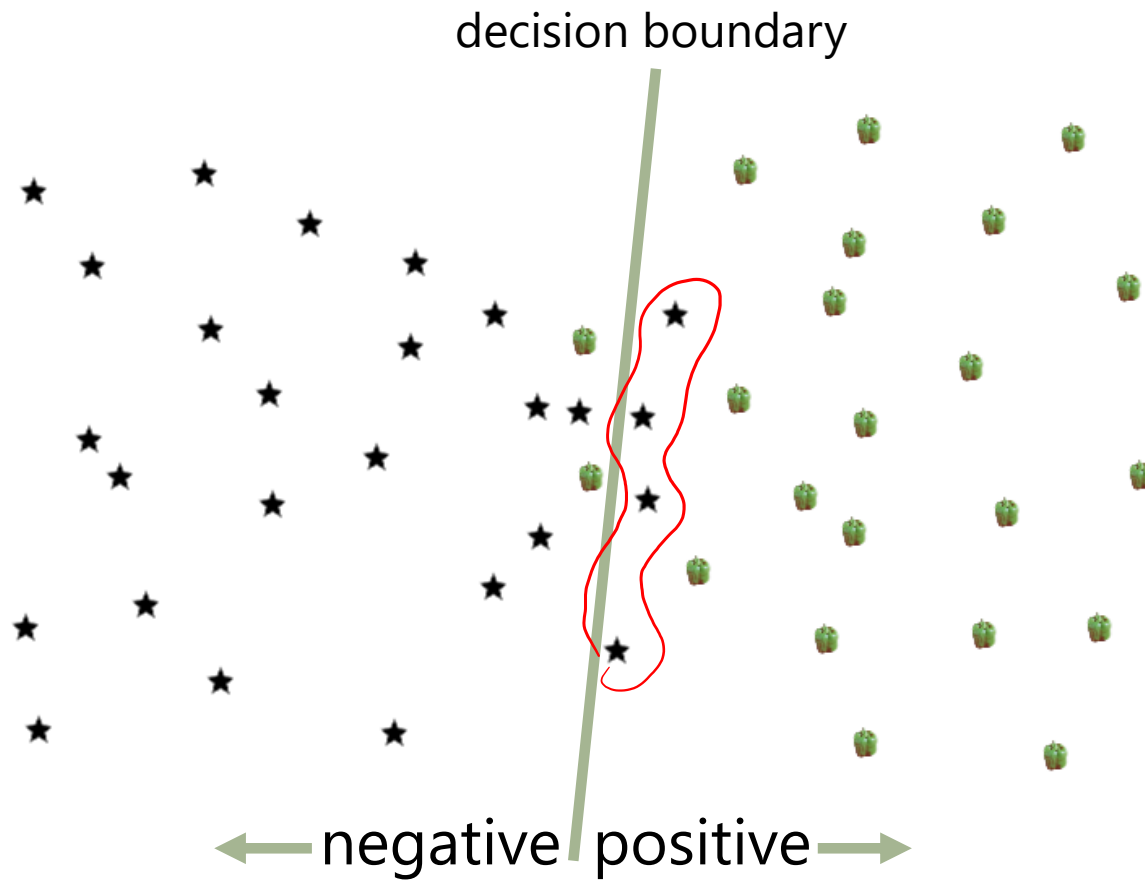
TP (true positive): Labeled as T , predicted as T

Evaluating classifiers



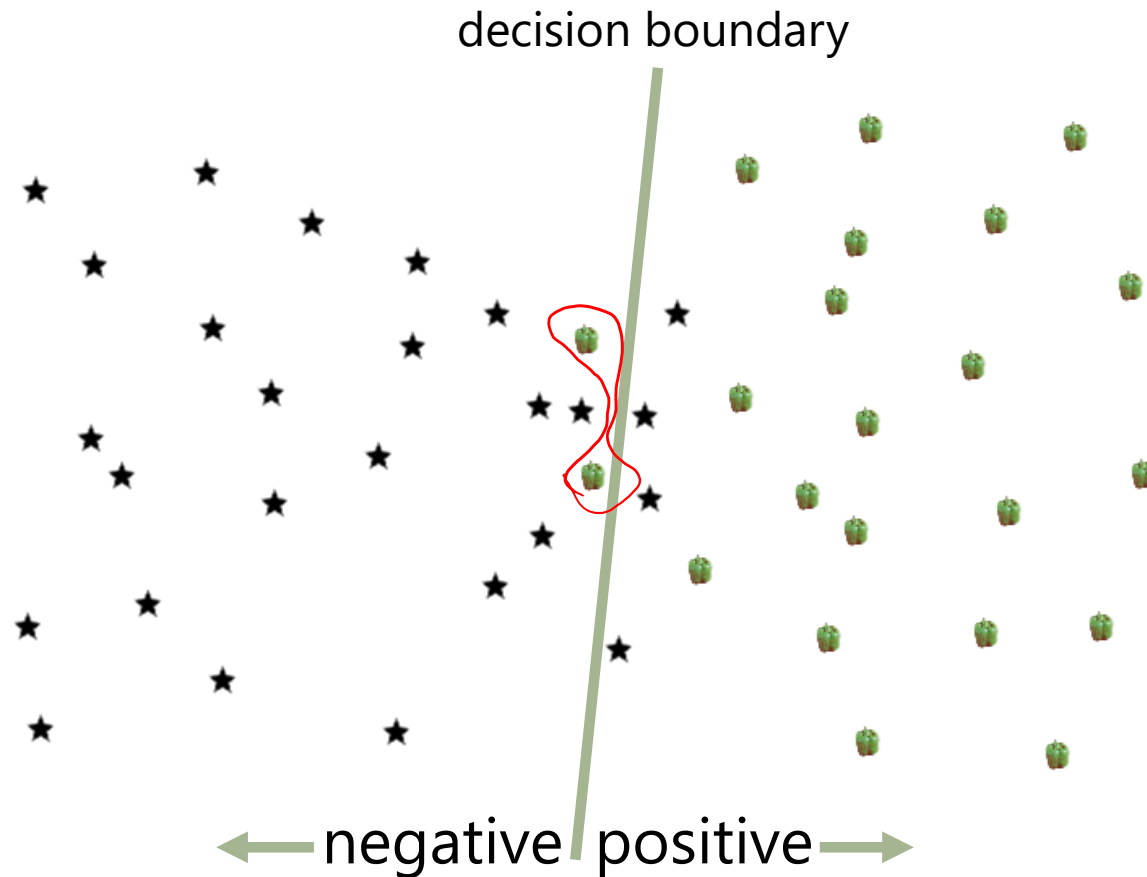
TN (true negative): Labeled as F , predicted as F

Evaluating classifiers



FP (false positive): Labeled as F , predicted as T

Evaluating classifiers



FN (false negative): Labeled as T , predicted as F

Evaluating classifiers

		Label	
		true	false
Prediction	true	true positive	false positive
	false	false negative	true negative

Classification accuracy = correct predictions / #predictions
= $(TP + TN) / (TP + FP + FN + TN)$

Error rate $1 - \text{acc}$ = incorrect predictions / #predictions
= $(FP + FN) / (TP + FP + TN + FN)$

Evaluating classifiers

		Label	
		true	false
Prediction	true	true positive	false positive
	false	false negative	true negative

True positive rate (**TPR**) = true positives / #labeled positive
= $TP / (TP + FN)$

True negative rate (**TNR**) = true negatives / #labeled negative
= $TN / (TN + FP)$

Evaluating classifiers

		Label	
		true	false
Prediction	true	true positive	false positive
	false	false negative	true negative

Balanced Error Rate (BER) = $\frac{1}{2}$ (FPR + FNR)

= $\frac{1}{2}$ for a random/naïve classifier, 0 for a perfect classifier

$$1 - \frac{1}{2} (TPR + TNR)$$

Evaluating classifiers

e.g.

$\mathbf{y} = [1, -1, 1, 1, 1, -1, 1, 1, -1, 1]$

Confidence = $[1.3, -0.2, -0.1, -0.4, 1.4, 0.1, 0.8, 0.6, -0.8, 1.0]$

\uparrow
 $X_i = 0$

tp ta fn fn tp fp tp tp ta tp

$$TPR = \frac{5}{7}$$

$$TNR = \frac{2}{3}$$

$$BER = 1 - \frac{1}{2} \left(\frac{5}{7} + \frac{2}{3} \right)$$

Evaluating classifiers

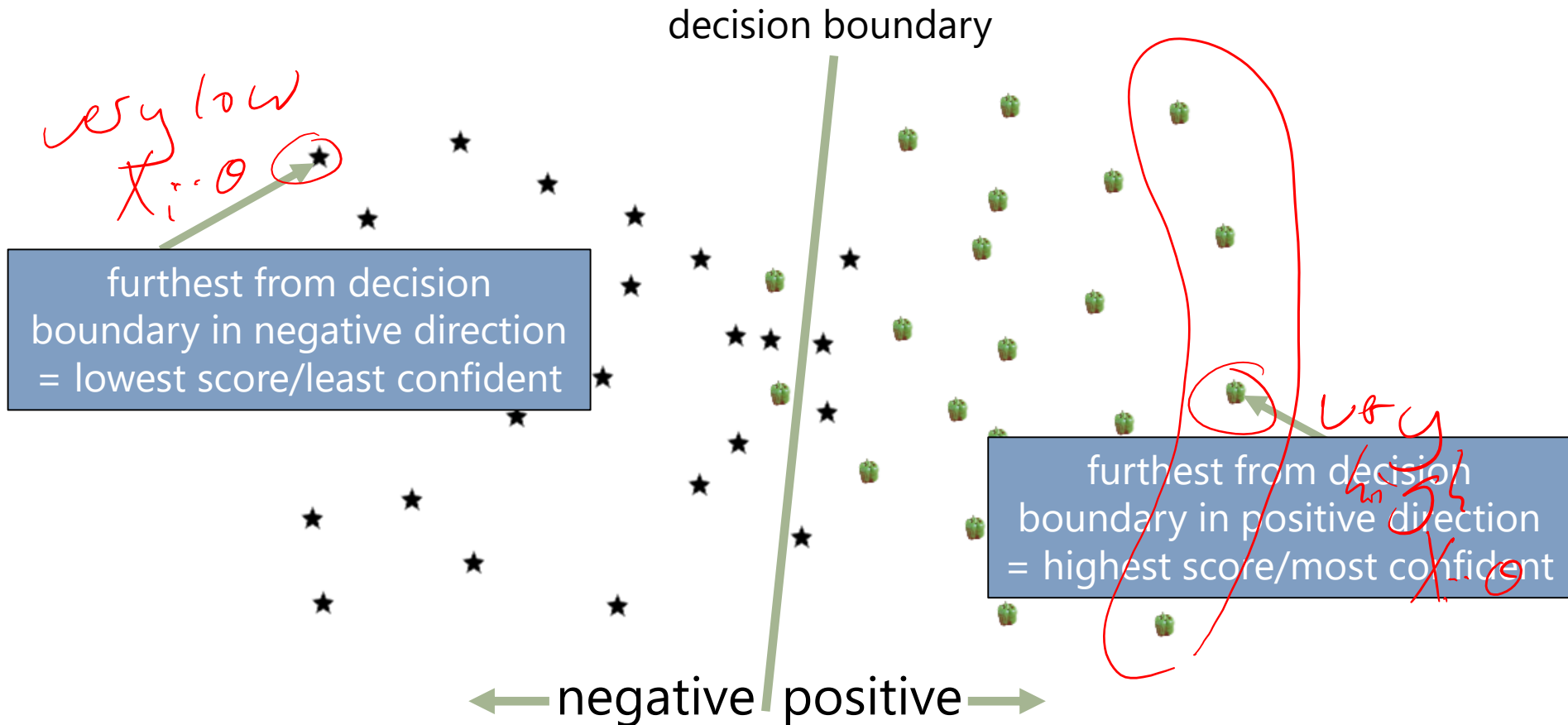
How to optimize a balanced error measure:

$$L_{\theta}(y|X) = \prod_{y_i=1} p_{\theta}(y_i|X_i) \prod_{y_i=0} (1 - p_{\theta}(y_i|X_i))$$

$$l_{\theta} = \frac{N}{2} \sum_{y_i=1} \sigma(X_i \cdot \theta) + \frac{N}{2} \sum_{y_i=0} (1 - \sigma(X_i \cdot \theta))$$

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction



Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

- In ranking settings, the actual labels assigned to the points (i.e., which side of the decision boundary they lie on) **don't matter**
- All that matters is that positively labeled points tend to be at **higher ranks** than negative ones

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

- For naïve Bayes, the "score" is the ratio between an item having a positive or negative class
- For logistic regression, the "score" is just the probability associated with the label being 1
- For Support Vector Machines, the score is the distance of the item from the decision boundary (together with the sign indicating what side it's on)

$-X_i - \theta$

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

e.g.

$$\mathbf{y} = [1, -1, 1, 1, 1, -1, 1, 1, -1, 1]$$

$$\mathbf{Confidence} = [1.3, -0.2, -0.1, -0.4, 1.4, 0.1, 0.8, 0.6, -0.8, 1.0]$$

Sort **both** according to confidence:


$$\text{conf} [1.4, 1.3, 1.0, 0.8, 0.6, 0.1, -0.1, -0.2, -0.4, -0.8]$$
$$\tilde{\mathbf{y}} = [1, 1, 1, 1, 1, -1, 1, -1, 1, -1]$$

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

Labels sorted by confidence:

[1, 1, 1, 1, 1, -1, 1, -1, 1, -1]



Suppose we have a fixed budget (say, six) of items that we can return (e.g. we have space for six results in an interface)

- Total number of **relevant** items = 7
- Number of items we returned = 6
- Number of **relevant items** we returned = 5

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

“fraction of retrieved documents that are relevant”

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

“fraction of relevant documents that were retrieved”

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

$\text{precision@}k$ = precision when we have a budget of k retrieved documents

e.g.

- Total number of **relevant** items = 7
- Number of items we returned = 6
- Number of **relevant items** we returned = 5

$$\text{precision@}6 = \frac{5}{6}$$

Evaluating classifiers – ranking

The classifiers we've seen can associate **scores** with each prediction

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

(harmonic mean of precision and recall)

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \text{precision} + \text{recall}}$$

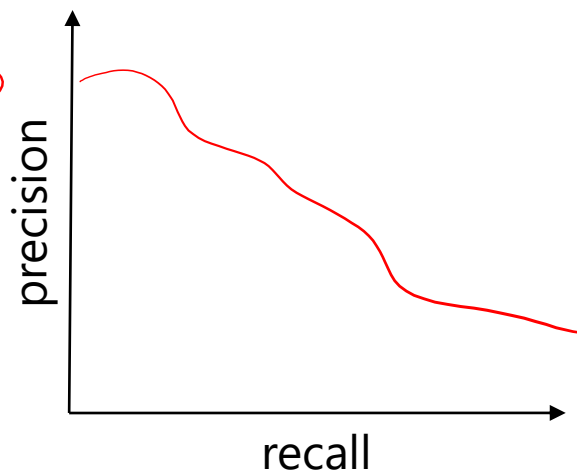
(weighted, in case precision is more important (low beta), or recall is more important (high beta))

Precision/recall curves

How does our classifier behave as we “increase the budget” of the number retrieved items?

- For budgets of size 1 to N, compute the precision and recall
- Plot the precision against the recall

*only returned
most confident
pt*



return everything

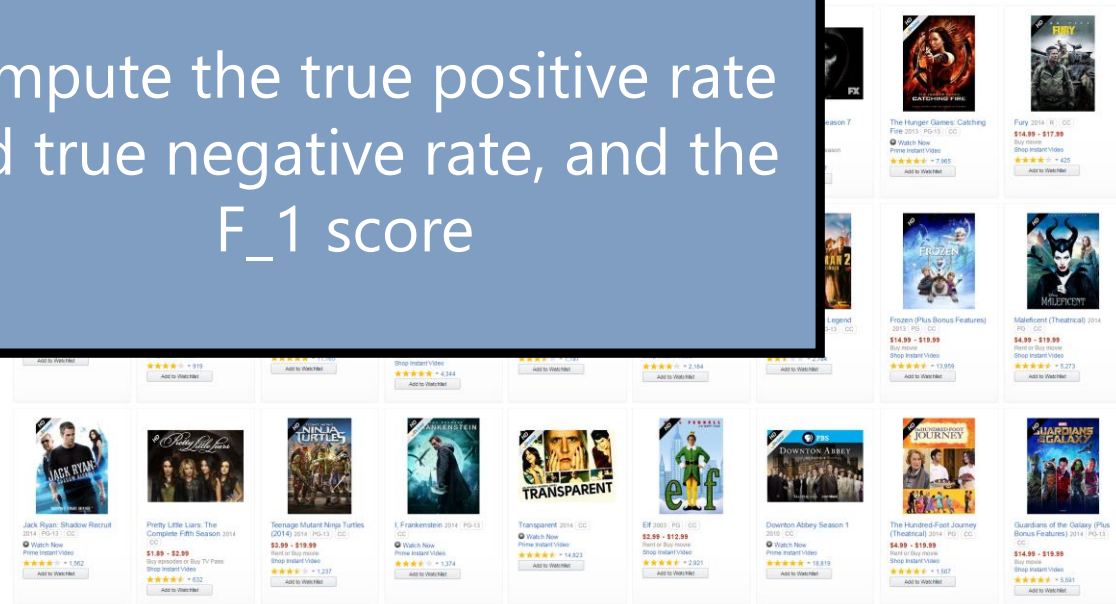
Summary

1. When data are highly imbalanced

If there are far fewer positive examples than negative examples we may want to assign additional weight to negative instances (or vice versa)

e.g. will I purchase product? If I purchase 0.00001% of products, then a classifier which just predicts "no" everywhere is 99.99999% accurate, but not very useful

Compute the true positive rate and true negative rate, and the F₁ score

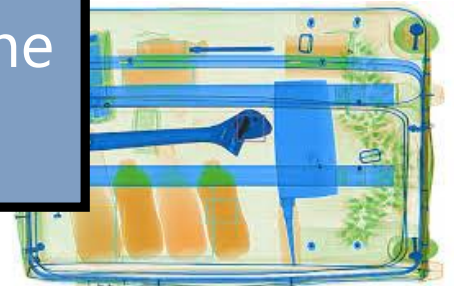
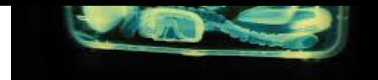


Summary

2. When mistakes are more costly in one direction

False positives are nuisances but false negatives are disastrous (or vice versa)

Compute “weighted” error measures that trade-off the precision and the recall, like the F_{β} score



e.g. which of these bags contains a weapon?

Summary

3. When we only care about the “most confident” predictions

e.g. does
result
among
page of results?

Compute the precision@k, and
plot the signature of precision
versus recall

tea station

Search tools

in the ... We'd like
port.

676 Reviews of Tea Station "Taro tea with boba was soooo good! Great service, too! The shaved ice is very good at a reasonable price too."

[Tea Station - Mira Mesa - San Diego, CA | Yelp](#)
www.yelp.com › Restaurants › Taiwanese ▾ Yelp ▾
★★★★☆ Rating: 3 - 381 reviews - Price range: \$
381 Reviews of Tea Station "Yes, I agree with Messiah! Everything is expensive but honestly the teas and boba are really delicious! But expect to wait long, the ..."

[Tea Station - Artesia, CA | Yelp](#)
www.yelp.com › Food › Desserts ▾ Yelp ▾
★★★★☆ Rating: 3.5 - 494 reviews - Price range: \$
494 Reviews of Tea Station "Came here at 12am SUPER hungry after not eating dinner. I was afraid the kitchen was going to be closed since they close at 1 am."

So far: Regression

Product Details

Genres	Science Fiction, Action, Horror
Director	David Twohy
Starring	Vin Diesel, Radha Mitchell
Supporting actors	Cole Hauser, Keith David, Lewis Fitz-Gerald, Claudia Black, Rhiana Gr Angela Moore, Peter Chiang, Ken Twohy
Studio	NBC Universal
MPAA rating	R (Restricted)
Captions and subtitles	English Details ▾
Rental rights	24 hour viewing period. Details ▾
Purchase rights	Stream instantly and download to 2 locations Details ▾
Format	Amazon Instant Video (streaming online video and digital download)

A. Phillips

Reviewer ranking: #17,230,554

90% helpful
votes received on reviews
(151 of 167)

ABOUT ME
Enjoy the reviews...

ACTIVITIES
Reviews (16)
Public Wish List (2)
Listmania Lists (2)
Tagged Items (1)

HipCzech

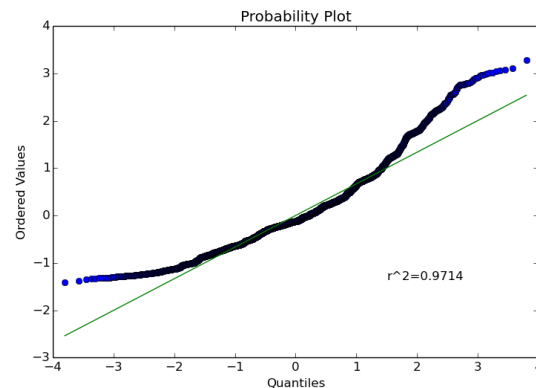
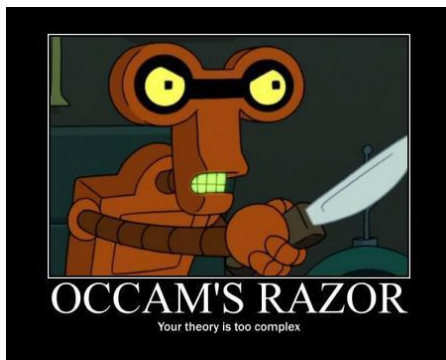
Aficionado
Male, from Texas

Profile Page

Member Since:	Jul 12, 2014	HipCzech was last seen:
Points:	175	Today at 12:19 AM
Beers:	108	
Places:	6	
Posts:	smoother than all of	0
Likes Received:	0	
Trading:	0% 0	

How can we use **features** such as product properties and user demographics to make predictions about **real-valued** outcomes (e.g. star ratings)?

How can we prevent our models from **overfitting** by favouring simpler models over more complex ones?



How can we assess our decision to optimize a particular error measure, like the MSE?

So far: Classification

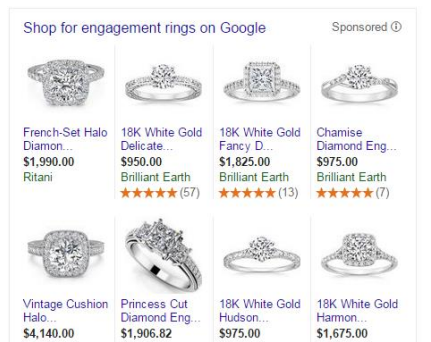
Next we adapted these ideas to **binary** or **multiclass** outputs



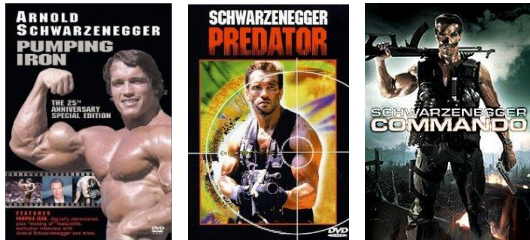
What animal is in this image?



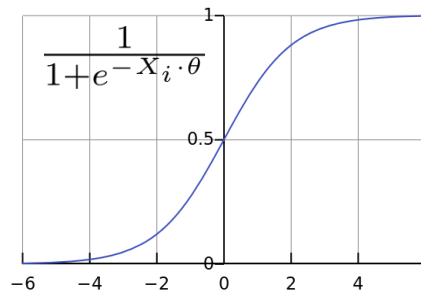
Will I **purchase** this product?



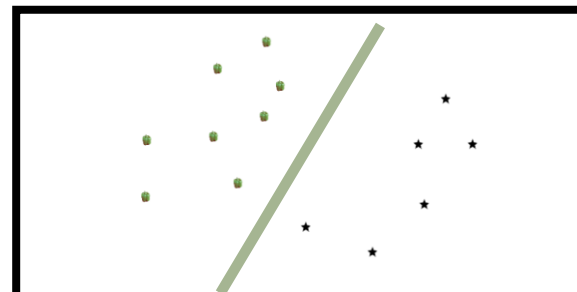
Will I **click on** this ad?



Combining features using naïve Bayes models



Logistic regression



Support vector machines

So far: supervised learning

Given **labeled training data** of the form

$\{(\text{data}_1, \text{label}_1), \dots, (\text{data}_n, \text{label}_n)\}$


Infer the function

$f(\text{data}) \xrightarrow{?} \text{labels}$

So far: supervised learning

We've looked at two types of prediction algorithms:

Regression  $y_i = X_i \cdot \theta$

Classification  $y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$

Questions?

Further reading:

- “Cheat sheet” of performance evaluation measures:
<http://www.damienfrancois.be/blog/files/modelperfcheatsheet.pdf>
 - Andrew Zisserman’s SVM slides, focused on
computer vision:
<http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf>