

CS 260: Foundations of Data Science

Prof. Thao Nguyen

Fall 2024



HVERFORD
COLLEGE

Admin

- **Lab 2** grades & feedback will be posted on Wednesday
- **Lab 3** due tonight
- **Lab 4** posted, due next Monday at midnight
- **Lecture Notes**

Peer Tutoring

- **Student tutors** (Fejiro Anigbro, Darshan Mehta)
- **Flexible hours**
- **Free!**

TECH TALKS 2024

OCTOBER 7,8 & 9TH | 6-8PM EST

***Sign up for a 30 minute virtual informational interview
with a Tri-Co alum to gain tech career insights!***

Alumni will represent various tech roles including software engineering and development, data science, tech consulting, product management and biotech.

OCT 7	OCT 8	OCT 9
Accenture ● FERMAT Commerce ● Grubhub	Bristol Myers Squibb ● Community.com ● C3 Presents (Live Nation) ● Opower (Oracle)	The Walt Disney Company ● Fresh Tracks Insights ● Meta ● Grubhub

TRI-COLLEGE RECRUITING CONSORTIUM

HAVERFORD

BRYN MAWR

SWARTHMORE

Outline for today

- Recap SGD (stochastic gradient descent)
- Introduction to classification
 - Decision tree models
 - Probabilistic interpretation
- Evaluation Metrics
 - Confusion matrices
 - Precision and recall
 - ROC curves

Outline for today

- Recap SGD (stochastic gradient descent)
- Introduction to classification
 - Decision tree models
 - Probabilistic interpretation
- Evaluation Metrics
 - Confusion matrices
 - Precision and recall
 - ROC curves

Stochastic Gradient Descent for Linear Regression

Key Idea: take the derivative of **one datapoint** at a time and use that to update w

$$\nabla J = \begin{bmatrix} \frac{\partial J}{\partial w_0} \\ \vdots \\ \frac{\partial J}{\partial w_p} \end{bmatrix}$$

Handout 6
142

derivative
wrt
example

$$J(\vec{w}) = \frac{1}{2} \sum_{i=1}^n (\underbrace{\vec{w} \cdot \vec{x}_i}_{\text{pred}} - \underbrace{y_i}_{\text{truth}})^2$$

derivative is very large + unstable

$$\nabla J(\vec{w})_{\vec{x}_i} = (\vec{w} \cdot \vec{x}_i - y_i) \vec{x}_i$$

datapoint scalar vector

$$\begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{bmatrix}$$

Stochastic Gradient Descent for Linear Regression

SGD

Start
while
for

for linear regression
with $\vec{w} = \vec{0}$ (vector of zeros)
(epoch) iteration t :

for $i = 1, 2, \dots, n$: (shuffle)

$$\vec{w} \leftarrow \vec{w} - \alpha (\vec{w} \cdot \vec{x}_i - y_i) \vec{x}_i$$

Step size derivative

check convergence
if

$$|J(\vec{w}^t) - J(\vec{w}^{t+1})| < \epsilon \Rightarrow \text{Stop!}$$

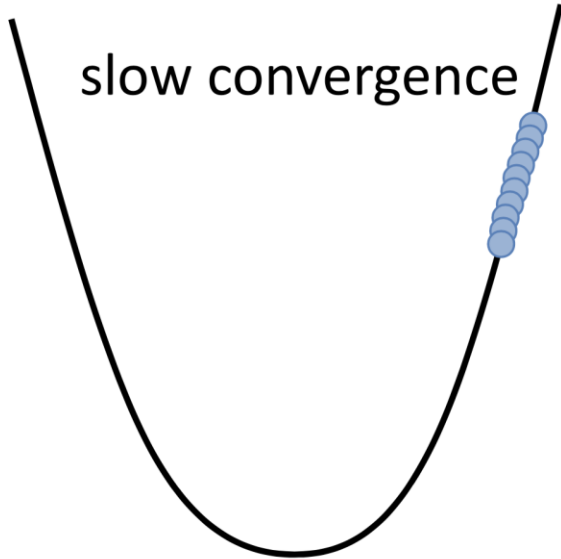
ϵ is very small

not for Lab 3

Choosing the step size alpha

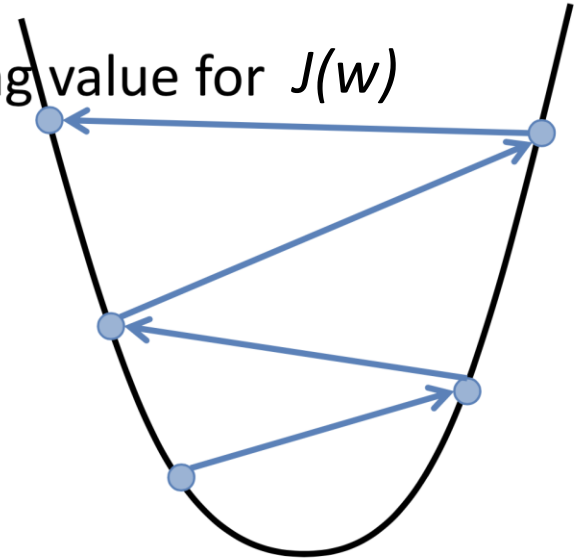
α too small

slow convergence



α too large

increasing value for $J(w)$



- may overshoot minimum
- may fail to converge (may even diverge)

Pros and Cons

(Analytic Solution)

Gradient Descent

- requires multiple iterations
- need to choose α
- works well when p is large
- can support online learning

Normal Equations

- non-iterative
- no need for α
- slow if p is large
 - matrix inversion is $O(p^3)$

Outline for today

- Recap SGD (stochastic gradient descent)
- Introduction to classification
 - Decision tree models
 - Probabilistic interpretation
- Evaluation Metrics
 - Confusion matrices
 - Precision and recall
 - ROC curves

Binary classification examples

- Transactions that indicate credit card fraud
- Accounts that are bots
- Detecting which scans show tumors
- Prenatal test for Down's Syndrome
- Finding genes under natural selection
- Regions of the environment that contains the object the robot is searching for

In all these examples, we are trying to find unusual items ("needle in a haystack") -- we call these *positives*

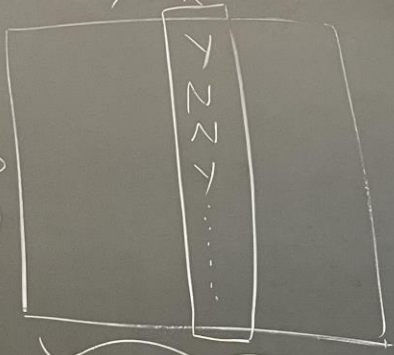
Introduction to Classification

Classification

X fever

Y (disease)

n examples



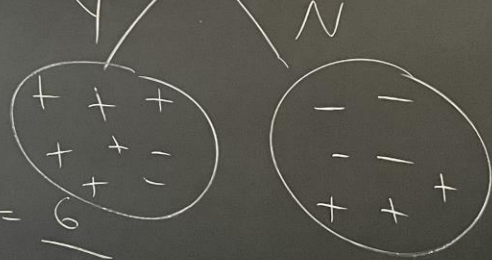
p features

training data

+ \Rightarrow disease
 - \Rightarrow no disease

Model: decision tree with a single feature ("stump")

fever



P_{pos} = prob of positive Y

$$P_{pos} = \frac{6}{8}$$

$$P_{pos} = \frac{3}{7}$$

n=15

Introduction to Classification

new idea : use probabilities
to classify test examples

$$\vec{x}_{\text{test}} = [\dots \text{fever} \dots]^T$$

threshold 0.5 \Rightarrow $\hat{y}_{\text{test}} = \ominus$ no disease

threshold 0.25 \Rightarrow $\hat{y}_{\text{test}} = \oplus$ disease

$P_{\text{pos}} \geq \text{threshold} \Rightarrow \text{classify } \oplus$

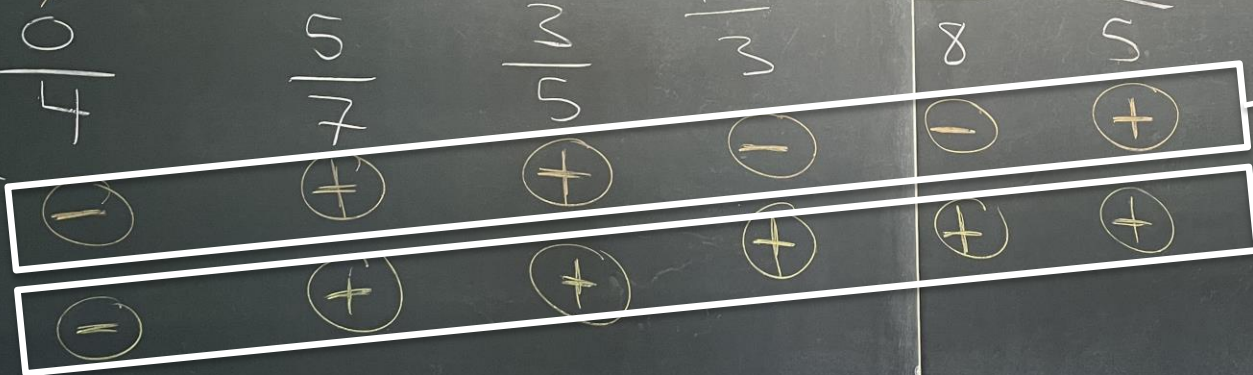
Handout 7

Handout 7

Slope

up flat down

bal
pos



thai

fixed normal reverse

$P_{pos} \geq 0.5 \Rightarrow pos$

$P_{pos} \geq \frac{1}{12} \Rightarrow pos$

Outline for today

- Recap SGD (stochastic gradient descent)
- Introduction to classification
 - Decision tree models
 - Probabilistic interpretation
- **Evaluation Metrics**
 - Confusion matrices
 - Precision and recall
 - ROC curves

Goals of Evaluation

- Think about what metrics are important for the problem at hand
- Compare different methods or models on the same problem
- Common set of tools that other researchers/users can understand

Training and Testing

(high-level idea)

- **Separate** data into “**train**” and “**test**”
 - n = num training examples
 - m = num testing examples
- **Fit** (create) the model using **training data**
 - e.g. sea_ice_1979-2012.csv
- **Evaluate** the model using **testing data**
 - e.g. sea_ice_2013-2020.csv

$$\frac{65+13}{100} = 78\% \text{ Pred}$$

	-	+
truth -	65	15
truth +	7	13

Accuracy =

Thresh = 0.5

test data

m = 100

50	30
1	19

Thresh 0.25

acc = 69%

80 negatives
20 positives

76	4
11	9

Thresh 0.75

correct

$$\frac{1}{m} \sum_{i=1}^m \mathbb{1}(\hat{y}_i = y_i)$$

Note: all the same model, different thresholds!



Confusion Matrices

Predicted class

Negative

Positive

Negative

True negative
(TN)

False positive
(FP)

True
class

Positive

False negative
(FN)

True positive
(TP)

Confusion Matrices

Predicted class

Negative

Positive

Negative

True negative (TN)	False positive (FP) “false alarm”
False negative (FN) “miss”	True positive (TP)

N (total number of true negatives)

True
class

Positive

P (total number of true positives)

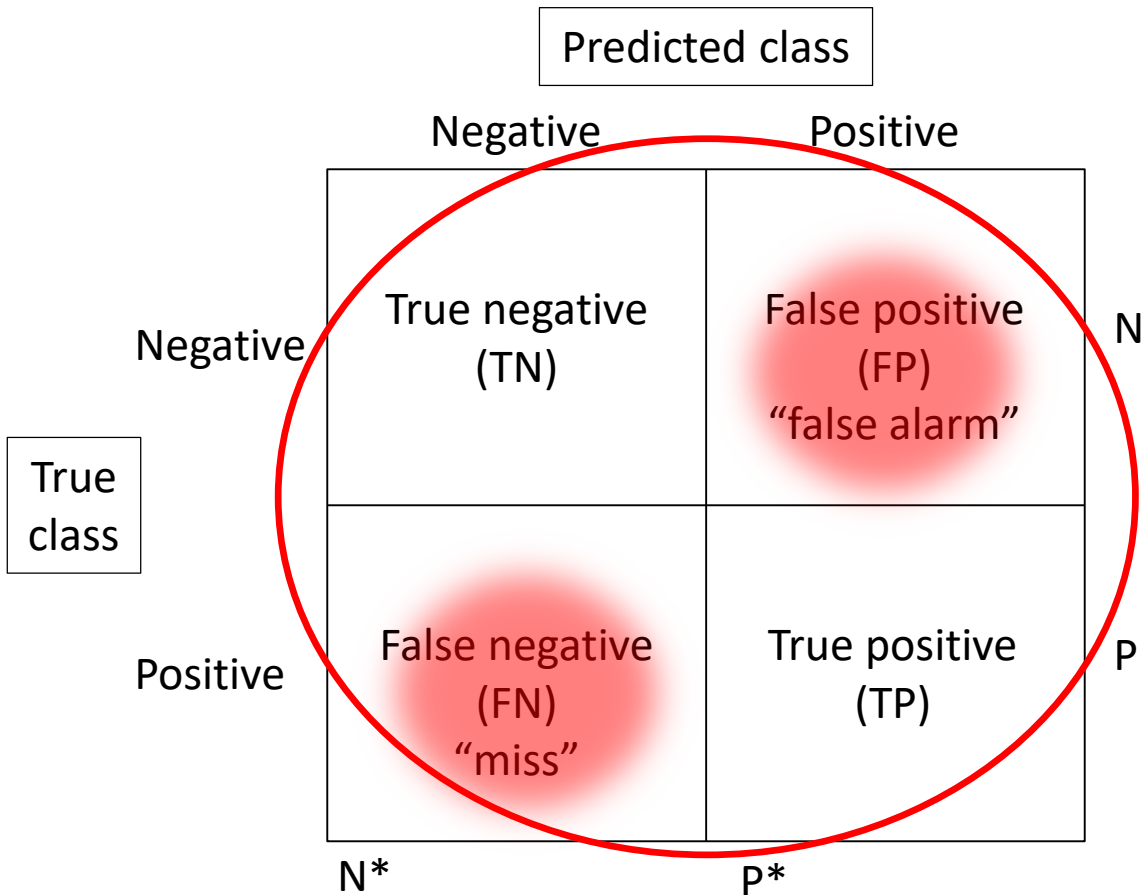
N* (what we said
was negative)

P* (what we said was
positive “flagged”)

Confusion Matrices

		Predicted class		
		Negative	Positive	
True class	Negative	True negative (TN) ✓	False positive (FP) "false alarm" ✗	N
	Positive	False negative (FN) "miss" ✗	True positive (TP) ✓	P
		N*	p*	

Confusion Matrices

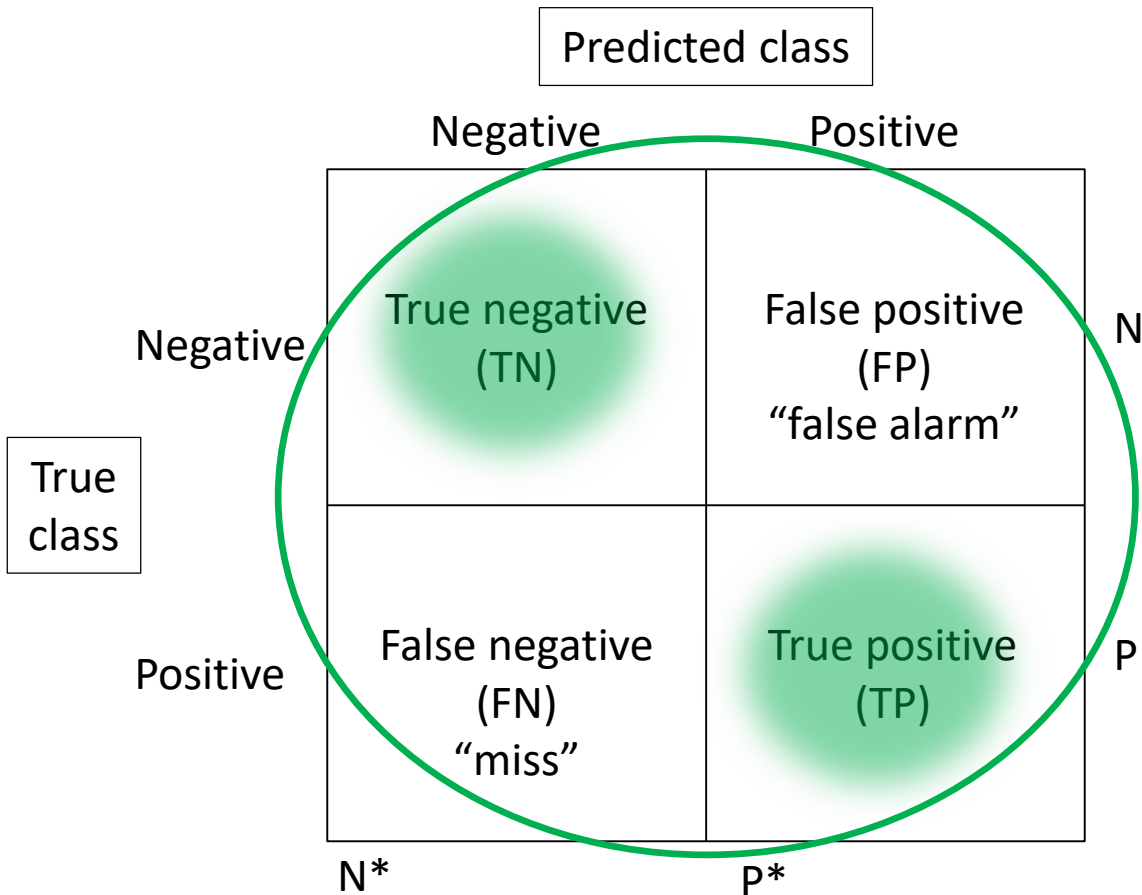


Error:

$$(FN+FP)/(TN+FP+FN+TP)$$

$$= (FN+FP)/(N+P)$$

Confusion Matrices



Accuracy = 1-Error:

$$(TN+TP)/(TN+FP+FN+TP)$$

$$= (TN+TP)/(N+P)$$

Confusion Matrices

		Predicted class	
		Negative	Positive
True class	Negative	True negative (TN)	False positive (FP) "false alarm"
	Positive	False negative (FN) "miss"	True positive (TP)
		N*	p*

The diagram shows a 2x2 confusion matrix. The columns represent the predicted class (Negative, Positive) and the rows represent the true class (Negative, Positive). The cells contain: True negative (TN), False positive (FP) "false alarm", False negative (FN) "miss", and True positive (TP). The matrix is annotated with marginal counts: N* for the total number of negative true cases, p* for the total number of positive predicted cases, N for the total number of negative predicted cases, and P for the total number of positive true cases. A purple oval highlights the FP and TP cells, and a purple gradient circle highlights the TP cell.

Precision:

$$TP / (FP + TP) = TP / P^*$$

Confusion Matrices

		Predicted class	
		Negative	Positive
True class	Negative	True negative (TN)	False positive (FP) "false alarm"
	Positive	False negative (FN) "miss"	True positive (TP)
		N*	p*

The diagram shows a 2x2 confusion matrix. The top row is labeled 'Negative' and the bottom row 'Positive' under the 'True class' label. The left column is labeled 'Negative' and the right column 'Positive' under the 'Predicted class' label. The cells contain: True negative (TN), False positive (FP) "false alarm", False negative (FN) "miss", and True positive (TP). The TP cell is highlighted with a blue oval. Marginal counts N* and p* are shown at the bottom, and N and P are shown on the right side of the matrix.

Recall
(True Positive Rate):

$$TP/(FN+TP) = TP/P$$

Confusion Matrices

		Predicted class	
		Negative	Positive
True class	Negative	True negative (TN)	False positive (FP) "false alarm"
	Positive	False negative (FN) "miss"	True positive (TP)
		N*	p*

The diagram shows a 2x2 confusion matrix. The top row is labeled 'Negative' and the bottom row 'Positive' under the 'True class' label. The left column is labeled 'Negative' and the right column 'Positive' under the 'Predicted class' label. The cells contain: True negative (TN), False positive (FP) "false alarm", False negative (FN) "miss", and True positive (TP). The total number of negative true class instances is N, and the total number of positive true class instances is p. The total number of negative predicted class instances is N*, and the total number of positive predicted class instances is p*.

False Positive Rate:

$$FP/(TN+FP) = FP/N$$

Precision and Recall

- Precision: of all the “flagged” examples, which ones are actually relevant (i.e. positive)?

(Purity)

- Recall: of all the relevant results, which ones did I actually return?

(Completeness)