CIS 419/519 Recitation

Varun Jana, Chang Liu

Sept 29/30 2020

Penn Engineering

Content

- Intuition Behind Linear Models, Learning
- Optimization, (S)GD, Mini Batch SGD
- Cross Validation Introduction
- Feature Extraction

Part 1: Linear Models & Learning

and the intuition

Fitting Lines & Training Models

- At a not-so-high-level, training models and fitting lines have a lot in common - training a model is like computing the best-fit line!
- So, model ~ line

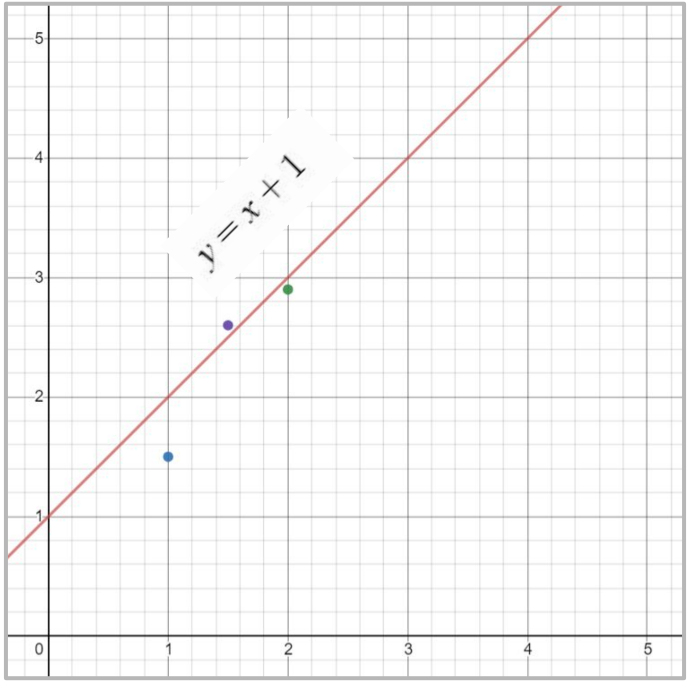
?

Fitting Lines & Training Models

Let's start with a simple line (2D):

Equation

$f(x) = y = \mathbf{a_1}x + \mathbf{a_0}$



Points

x (input)	y (output)
1	1.5
2	2.9
1.5	2.6

Find:

$\mathbf{A} = [a_0 \ a_1]$

Fitting Lines & Training Models

This is 'equivalent' to:

Model

$h_\Theta(x)$ = output label = $\Theta_1 x + \Theta_0$

Find

$\boldsymbol{\Theta} = [\, \Theta_0\ \Theta_1 \,]$

Instances

x (Feature 1)	$h_\Theta(x)$ **Output Label**
1	1.5
2	2.9
1.5	2.6

To mimic a classifier, simply map intervals to values!

Ex. $f(x) = \mathrm{sgn}(\boldsymbol{w}^T \cdot \boldsymbol{x} - \theta\} = \mathrm{sgn}\{\sum_{i=1}^{n} w_i x_i - \theta\}$

Into Higher Dimensions

‘Lines’ in higher dimensions ~ Linear models with many features

SEQN	RIDAGEYR	BMXWAIST	BMXHT	LBXTC	BMXLEG	BMXWT	BMXBMI	RIDRETH1	BPQ020	ALQ120Q	DMDEDUC2	RIAGENDR	INDFMPIR	LBXGH	DIABETIC
73557	69.0	100.0	171.3	167.0	39.2	78.3	26.7	Non-Hispanic Black	yes	1.0	high school graduate / GED	male	0.84	13.9	yes
73558	54.0	107.6	176.8	170.0	40.0	89.5	28.6	Non-Hispanic White	yes	7.0	high school graduate / GED	male	1.78	9.1	yes
73559	72.0	109.2	175.3	126.0	40.0	88.9	28.9	Non-Hispanic White	yes	0.0	some college or AA degree	male	4.51	8.9	yes
73562	56.0	123.1	158.7	226.0	34.2	105.0	41.7	Mexican American	yes	5.0	some college or AA degree	male	4.79	5.5	no
73564	61.0	110.8	161.8	168.0	37.1	93.4	35.7	Non-Hispanic White	yes	2.0	college graduate or above	female	5.0	5.5	no
73566	56.0	85.5	152.8	278.0	32.4	61.8	26.5	Non-Hispanic White	no	1.0	high school graduate / GED	female	0.48	5.4	no
73567	65.0	93.7	172.4	173.0	40.0	65.3	22.0	Non-Hispanic White	no	4.0	9th-11th grade	male	1.2	5.2	no
73568	26.0	73.7	152.5	168.0	34.4	47.1	20.3	Non-Hispanic White	no	2.0	college graduate or above	female	5.0	5.2	no
73571	76.0	122.1	172.5	167.0	35.5	102.4	34.4	Non-Hispanic White	yes	2.0	college graduate or above	male	5.0	6.9	yes
73577	32.0	100.0	166.2	182.0	36.5	79.7	[illegible]	[illegible] American	no	20.0	Less than 9th grade	male	0.29	5.3	no
73581	50.0	99.3	185.0	202.0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	5.0	5.0	no
73585	28.0	90.3	175.1	198.0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2.26	5.0	no
73589	35.0	94.6	172.9	192.0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1.74	5.5	no
73595	58.0	114.8	175.3	165.0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	3.09	7.7	no
73596	57.0	117.8	164.7	151.0	35.3	104.0	38.3	Other or Multi-Racial	yes	1.0	college graduate or above	female	5.0	5.9	no
73600	37.0	122.9	185.1	189.0	48.1	126.2	36.8	Non-Hispanic Black	yes	2.0	high school graduate / GED	male	0.63	[illegible]	yes
73604	69.0	96.6	156.9	203.0	37.0	59.5	24.2	Non-Hispanic White	no	1.0	some college or AA degree	female	[illegible]	5.4	no
73607	75.0	130.5	169.6	161.0	36.5	111.9	38.9	Non-Hispanic White	yes	0.0	high school graduate / GED	male	[illegible]	[illegible]	no
73610	43.0	102.6	176.8	200.0	38.8	90.2	28.9	Non-Hispanic White	no	5.0	college graduate or above	male	[illegible]	[illegible]	no
73613	60.0	113.6	163.8	203.0	41.6	104.9	39.1	Non-Hispanic Black	yes	2.0	9th-11th grade	[illegible]	[illegible]	[illegible]	no
73614	55.0	90.9	167.9	256.0	43.5	60.9	21.6	Non-Hispanic White	no	0.0	high school graduate / GED	[illegible]	[illegible]	[illegible]	no
73615	65.0	100.3	145.9	166.0	30.0	55.4	26.0	Other Hispanic	yes	1.0	Less than 9th grade	female	1.22	6.3	yes

rows denote labeled instances $\langle \boldsymbol{x}_i, y_i \rangle$

classes

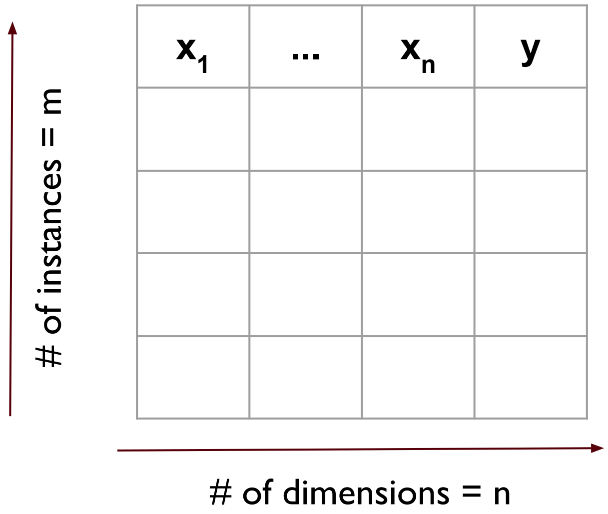


image source: CIS 519 Spring ‘20

Into Higher Dimensions

'Lines' in higher dimensions ~ Linear models with many features

A More 'Complex' Model

$$h_\Theta(x) = \text{output label} = \Theta_n + \ldots + \Theta_1 x + \Theta_0$$

Find

$$\boldsymbol{\Theta} = [\ \Theta_0\ \Theta_1\ \ldots\ \Theta_n\ ]$$

Penn Engineering

The Equivalencies

So:

- Line ~ Model
- $f(x)$ ~ $h_\Theta(x)$
- Coefficients in function ~ Weights of classifier (“linear” - like a line; ‘linear’ coefficients + c)
- Dimension ~ Feature (“high dimensionality” ~ more features)
- Intercept ~ Θ_0

Errors & Costs

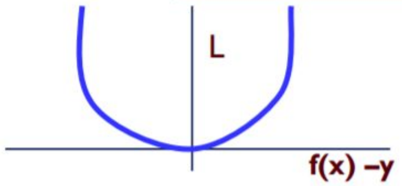
- A Loss Function $L(f(\boldsymbol{x}), y)$ measure how far is the prediction f(x) from the desired y; ~ **how far points are from the best fit line**
  - the penalty incurred by a classifier f on example $(\boldsymbol{x}, y)$.
- There are many different loss functions one could define:
  - Misclassification Error: (0-1 loss)

    $L(f(\boldsymbol{x}), y) = 0$ if $f(\boldsymbol{x}) = y$; 1 otherwise
  - Squared Loss:

    $L(f(\boldsymbol{x}), y) = (f(\boldsymbol{x}) - y)^2$
  - Input dependent loss:

    $L(f(\boldsymbol{x}), y) = 0$ if $f(\boldsymbol{x}) = y$; $c(\boldsymbol{x})$otherwise.

A continuous convex loss function allows a simpler optimization algorithm.



Part 2: (S)GD, Mini-Batch GD

Gradient Descent - Illustration

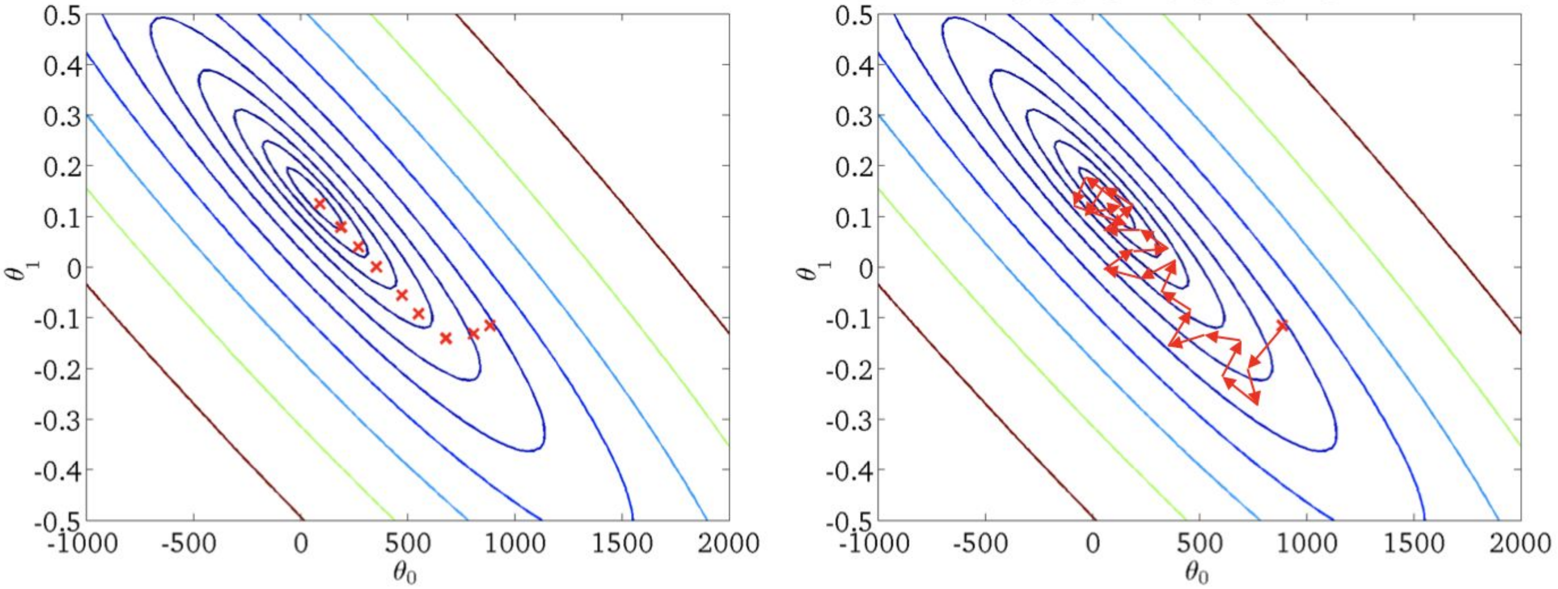
Learning outcomes:

- Linear Algebra, Optimization Calculus + intuition behind equations
- How this works
- Hyperparameters, variable learning rates
- Implementation

We will use the following article for this (it includes code with great visualizations so you can see how the weight vector updates etc). I highly recommend you go over this separately as well!

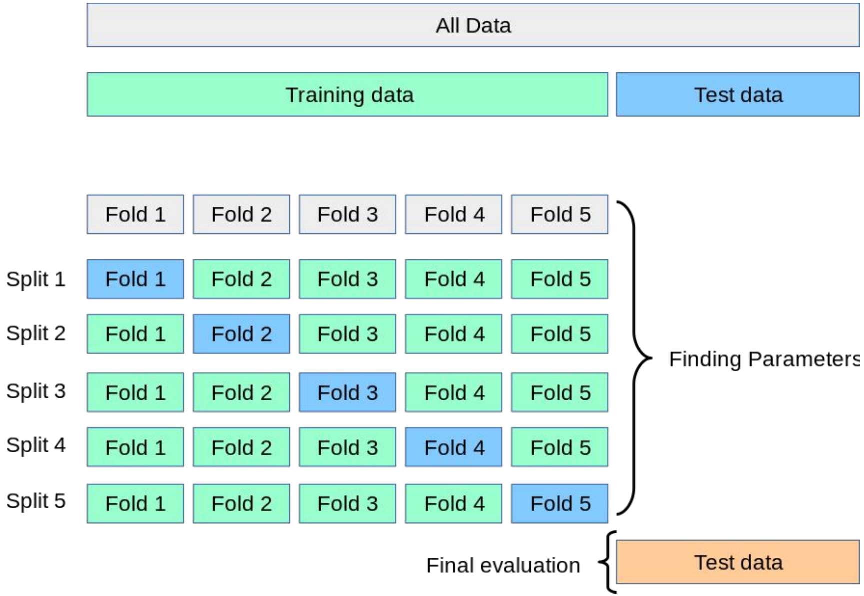
https://towardsdatascience.com/gradient-descent-in-python-a0d07285742f

GD vs SGD


0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-1000
-500
0
500
1000
1500
2000
θ_0
θ_1

Part 3: Cross Validation

Cross Validation : Concept



The scikit-learn docs give you not only the **code** but also explanations on this.
Check it out! https://scikit-learn.org/stable/modules/cross_validation.html

image source: scikit-learn

The Need for CV

- Train robust models by “creating more” data
- Compare models the ‘black box’ way
- Evaluate (significance tests)

Part 4: Feature Extraction (HW1)

One-Hot-Encoding

Will I Go to Play Today?

feature

an instance

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

binary label

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

Will I Go to Play Today?

	O	T	H	W	Play?
1	S	H	H	0	0
2	S	H	H	1	0
3	O	H	H	0	1
4	R	M	H	0	1
5	R	C	N	0	1
6	R	C	N	1	0
7	O	C	N	1	1
8	S	M	H	0	0
9	S	C	N	0	1
10	R	M	N	0	1
11	S	M	N	1	1
12	O	M	H	1	1
13	O	H	N	0	1
14	R	M	H	1	0

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

How to represent
Outlook / Temperature/ Humidity
Numerically?

Categorical Data

	O	T	H	W	Play?
1	S	H	H	0	0
2	S	H	H	1	0
3	O	H	H	0	1
4	R	M	H	0	1
5	R	C	N	0	1
6	R	C	N	1	0
7	O	C	N	1	1
8	S	M	H	0	0
9	S	C	N	0	1
10	R	M	N	0	1
11	S	M	N	1	1
12	O	M	H	1	1
13	O	H	N	0	1
14	R	M	H	1	0

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

Categorical data are variables that contain label values rather than numeric values. The number of possible values is often limited to a fixed set.

How to convert to numerical data?

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

Integer Encoding

One-Hot Encoding

O	O	O_s	O_o	O_r
S	0	1	0	0
S	0	1	0	0
O	1	0	1	0
R	2	0	0	1
R	2	0	0	1
R	2	0	0	1
O	1	0	1	0
S	0	1	0	0
S	0	1	0	0
R	2	0	0	1
S	0	1	0	0
O	1	0	1	0
O	1	0	1	0
R	2	0	0	1

S : 0
O : 1
R : 2

Penn Engineering

Numerical Features

O_s	O_o	O_r	T_h	T_m	T_c	H_h	H_n	H_l	W	Play?
1	0	0	1	0	0	1	0	0	0	0
1	0	0	1	0	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0	0	1
0	0	1	0	1	0	1	0	0	0	1
0	0	1	0	0	1	0	1	0	0	1
0	0	1	0	0	1	0	1	0	1	0
0	1	0	0	0	1	0	1	0	1	1
1	0	0	0	1	0	1	0	0	0	0
1	0	0	0	0	1	0	1	0	0	1
0	0	1	0	1	0	0	1	0	0	1
1	0	0	0	1	0	0	1	0	1	1
0	1	0	0	1	0	1	0	0	1	1
0	1	0	1	0	0	0	1	0	0	1
0	0	1	0	1	0	1	0	0	1	0

Penn Engineering

Useful Tools

1. Dictionary is a very useful tool for mapping in Python
2. pd.get_dummies()
3. pd.Categoricals()