

CS-GY 6923: Lecture 1

Introduction to Machine Learning

NYU Tandon School of Engineering, Prof. Christopher Musco

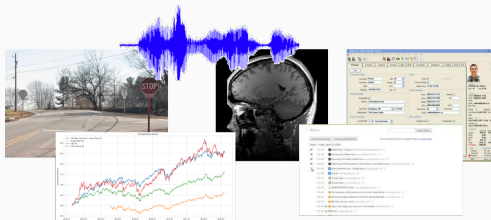
NO RIGHT ANSWERS

- What is **Machine Learning**?
- How is it different, the same as **Artificial Intelligence**?
- How is it different, the same as **Statistics**?

BASIC GOAL

Goal: Develop algorithms to make decisions or predictions based on data.

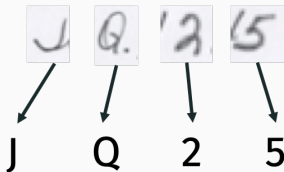
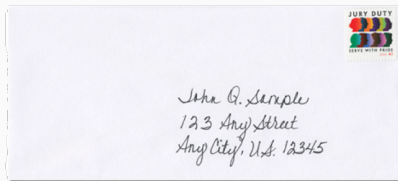
- **Input:** A single piece of data (an image, audio file, patient healthcare record, MRI scan).



- **Output:** A prediction or decision (this image is a stop sign, this stock will go up 10% next quarter, turn the car right).

CLASSIC EXAMPLE

Optical character recognition (OCR): Decide if a handwritten character is an $a, b, \dots, z, 0, 1, \dots, 9, \dots$



Optical character recognition (OCR): Decide if a handwritten character is an $a, b, \dots, z, 0, 1, \dots, 9, \dots$

Applications:

- Automatic mail sorting.
- Text search in handwritten documents.
- Digitizing scanned books.
- License plate detection for tolls.
- Etc.

How would you write an **algorithm** to distinguish these digits?



0 1 2 3 4 5 6 7 8 9

Suppose you just want to distinguish between a 1 and a 7.

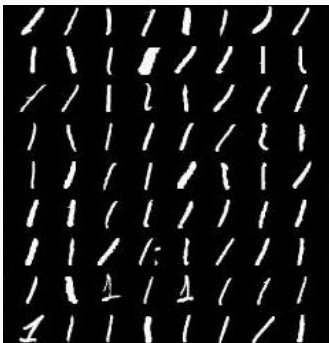
1S VS. 7S ALGORITHM

Reasonable approach: A number which contains one vertical line is a 1, if it contains one vertical and one horizontal line, it's a 7.

```
1  def count_vert_lines(image):
2  ...
3
4  def count_horiz_lines(image):
5  ...
6
7  def classify(image):
8  ...
9      nv = count_vert_lines(image)
10     nh = count_vert_lines(image)
11
12     if (nv == 1) and (nh == 1):
13         return '7'
14     elif (nv == 1) and (nh == 0):
15         return '1'
16     elif ...
```

1S VS. 7S ALGORITHM

This rule breaks down in practice:



Even fixes/modifications of the rule tend to be brittle... Maybe you could get 80% accuracy, but not nearly good enough.

Rule based systems, also called Expert Systems were the dominant approach to artificial intelligence in the 70s and 80s.

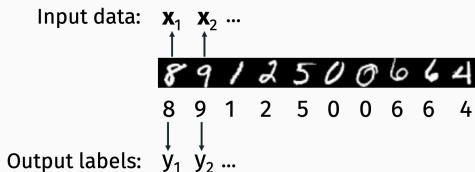
Major limitation: While human's are very good at many tasks,

- It's often hard to encode why humans make decisions in simple programmable logic.
- We think in abstract concepts with no mathematical definitions (how exactly do you define a line? how do you define a curve? straight line?)

A DIFFERENT APPROACH: MACHINE LEARNING

Focus on what humans do well: solving the task at hand!

Step 1: Collect and label many input/output pairs (x_i, y_i) . For our digit images, we have each $x_i \in \mathbb{R}^{28 \times 28}$ and $y_i \in \{0, 1, \dots, 9\}$.



This is called the **training dataset**.

Step 2: Learn from the examples we have.

- Have the computer automatically find some function $f(\mathbf{x})$ such that $f(\mathbf{x}_i) = y_i$ for most (\mathbf{x}_i, y_i) in our training data set (by searching over many possible functions).

Think of f as any crazy equation, or an arbitrary program:

$$f(\mathbf{x}) = 10 \cdot x[1, 1] - 6 \cdot x[3, 45] \cdot x[9, 99] + 5 \cdot \text{mean}(\mathbf{x}) + \dots$$

This approach of learning a function from labeled data is called **supervised learning**.

SUPERVISED LEARNING FOR OCR

National Institute for Standards and Technology collected a huge amount of handwritten digit data from census workers and high school students in the early 90s:

HANDWRITING SAMPLE FORM

NAME [REDACTED] DATE 8-2-89 CITY Memphis, TN STATE TN

The merge of handwriting being collected for use in testing computer recognition of hand printed numbers and letters. Please print the following characters in the boxes that appear below.

0123456789	0123456789	0123456789
01	02	03
04	05	06
07	08	09
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36
37	38	39
40	41	42
43	44	45
46	47	48
49	50	51
52	53	54
55	56	57
58	59	60
61	62	63
64	65	66
67	68	69
70	71	72
73	74	75
76	77	78
79	80	81
82	83	84
85	86	87
88	89	90
91	92	93
94	95	96
97	98	99
00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17
18	19	20
21	22	23
24	25	26
27	28	29
30	31	32
33	34	35
36	37	38
39	40	41
42	43	44
45	46	47
48	49	50
51	52	53
54	55	56
57	58	59
60	61	62
63	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	00	01
02	03	04
05	06	07
08	09	10
11	12	13
14	15	16
17	18	19
20	21	22
23	24	25
26	27	28
29	30	31
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	64
65	66	67
68	69	70
71	72	73
74	75	76
77	78	79
80	81	82
83	84	85
86	87	88
89	90	91
92	93	94
95	96	97
98	99	00
01	02	03
04	05	06
07	08	09
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36
37	38	39
40	41	42
43	44	45
46	47	48
49	50	51
52	53	54
55	56	57
58	59	60
61	62	63
64	65	66
67	68	69
70	71	72
73	74	75
76	77	78
79	80	81
82	83	84
85	86	87
88	89	90
91	92	93
94	95	96
97	98	99
00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17
18	19	20
21	22	23
24	25	26
27	28	29
30	31	32
33	34	35
36	37	38
39	40	41
42	43	44
45	46	47
48	49	50
51	52	53
54	55	56
57	58	59
60	61	62
63	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	00	01
02	03	04
05	06	07
08	09	10
11	12	13
14	15	16
17	18	19
20	21	22
23	24	25
26	27	28
29	30	31
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	64
65	66	67
68	69	70
71	72	73
74	75	76
77	78	79
80	81	82
83	84	85
86	87	88
89	90	91
92	93	94
95	96	97
98	99	00
01	02	03
04	05	06
07	08	09
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36
37	38	39
40	41	42
43	44	45
46	47	48
49	50	51
52	53	54
55	56	57
58	59	60
61	62	63
64	65	66
67	68	69
70	71	72
73	74	75
76	77	78
79	80	81
82	83	84
85	86	87
88	89	90
91	92	93
94	95	96
97	98	99
00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17
18	19	20
21	22	23
24	25	26
27	28	29
30	31	32
33	34	35
36	37	38
39	40	41
42	43	44
45	46	47
48	49	50
51	52	53
54	55	56
57	58	59
60	61	62
63	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	00	01
02	03	04
05	06	07
08	09	10
11	12	13
14	15	16
17	18	19
20	21	22
23	24	25
26	27	28
29	30	31
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	64
65	66	67
68	69	70
71	72	73
74	75	76
77	78	79
80	81	82
83	84	85
86	87	88
89	90	91
92	93	94
95	96	97
98	99	00
01	02	03
04	05	06
07	08	09
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36
37	38	39
40	41	42
43	44	45
46	47	48
49	50	51
52	53	54
55	56	57
58	59	60
61	62	63
64	65	66
67	68	69
70	71	72
73	74	75
76	77	78
79	80	81
82	83	84
85	86	87
88	89	90
91	92	93
94	95	96
97	98	99
00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17
18	19	20
21	22	23
24	25	26
27	28	29
30	31	32
33	34	35
36	37	38
39	40	41
42	43	44
45	46	47
48	49	50
51	52	53
54	55	56
57	58	59
60	61	62
63	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	00	01
02	03	04
05	06	07
08	09	10
11	12	13
14	15	16
17	18	19
20	21	22
23	24	25
26	27	28
29	30	31
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55
56	57	58
59	60	61
62	63	64
65	66	67
68	69	70
71	72	73
74	75	76
77	78	79
80	81	82
83	84	85
86	87	88
89	90	91
92	93	94
95	96	97
98	99	00
01	02	03
04	05	06
07	08	09
10	11	12
13	14	15
16	17	18
19	20	21
22	23	24
25	26	27
28	29	30
31	32	33
34	35	36
37	38	39
40	41	42
43	44	45
46	47	48
49	50	51
52	53	54
55	56	57
58	59	60
61	62	63
64	65	66
67	68	69
70	71	72
73	74	75
76	77	78
79	80	81
82	83	84
85	86	87
88	89	90
91	92	93
94	95	96
97	98	99
00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17
18	19	20
21	22	23
24	25	26
27	28	29
30	31	32
33	34	35
36	37	38
39	40	41
42	43	44
45	46	47
48	49	50
51	52	53
54	55	56
57	58	59
60	61	62
63	64	65
66	67	68
69	70	71
72	73	74
75	76	77
78	79	80
81	82	83
84	85	86
87	88	89
90	91	92
93	94	95
96	97	98
99	00	01
02	03	04
05	06	07
08	09	10
11	12	13
14	15	16
17	18	19
20	21	22
23	24	25
26	27	28
29	30	31
32	33	34
35	36	37
38	39	40
41	42	43
44	45	46
47	48	49
50	51	52
53	54	55

Since the 1990s machine learning have overtaken expert systems as the dominant approach to artificial intelligence.

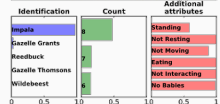
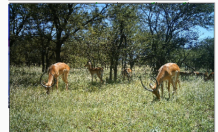
- Current methods achieve .17% error rate¹ for OCR on benchmark datasets (MNIST).²
- Very successful on other problems as well.

¹Last time I taught this course it was .21%.

²Not because of overfitting! See: *Cold Case: The Lost MNIST Digits* by Chhavi Yadav + Léon Bottou.

You could not be studying ML at a more exciting time!

- Autonomous vehicles.
- Human level play in very difficult games.
- Incredible machine translation.
- Pervasive impact in science and engineering.
- Many, many more.



WHAT IS DRIVING MACHINE LEARNING?

Machine learning has benefited from an explosion in our ability to collect and store data:

- Cheap, fast storage. Large data centers accessible via the cloud.
- Pervasive monitoring (satellite imagery, cheap sensors, improved and reduced cost for technologies like LIDAR).
- Crowd-sourced data collection (images, text on the internet)
- Crowd-sourced data labeling via the internet (Amazon Mechanical Turk, reCAPTCHA, etc.)

Having lots of data isn't enough. We have to know how to use it effectively.

Once we have the basic machine learning setup, many very difficult questions remain:

- How do we **parameterize** a class of functions f to search?
- How do we **efficiently find** a good function in the class?
- How do we ensure that an $f(\mathbf{x})$ which works well on our training data will **generalize** to perform well on future data?
- How do we deal with **imperfect data** (noise, outliers, incorrect training labels)?

In this course you will learn to answer these central questions through a combination of:

- Hands on implementation.
 - In-class demos and take-home labs using **Python** and **Jupyter notebooks**.
 - We will use **Google Colab** as the primary programming environment.
 - Mini-final project (on any dataset/problem you like).
- Theoretical exploration.
 - Written problem sets.
 - Midterm and final exam.

Goals of hands-on component:

1. Learn how to view and formulate real world problems in the language of machine learning.
2. Gain experience applying the most popular and successful machine learning algorithms to example problems.

Goals of theoretical component:

1. Learn how theoretical analysis can help explain the performance of machine learning algorithms and lead to the design of entirely new methods.
2. Build experience with the most important mathematical tools used in machine learning, including probability, statistics, and linear algebra. This experience will prepare you for more advanced coursework in ML, or research.
3. Be able to understand contemporary research in machine learning, including papers from NeurIPS, ICML, ICLR, and other major machine learning venues.

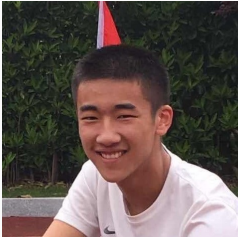
- CS-GY 6763: **Algorithmic Machine Learning and Data Science** (Dr. Rajesh Jayaram)
- CS-GY 9223: **Statistical and Computational Foundations of Machine Learning** (Dr. David Pal)
- CS-GY 9223: **Foundations of Deep Learning** (Prof. Chinmay Hegde, Ph.D. students only)

All class information can be found at:

www.chrismusco.com/machinelearning2022

TWO MOST IMPORTANT THINGS FROM SYLLABUS

1. Make sure you are signed up for **Ed Stem** which will be used for all classroom communication (no email).
2. Don't hesitate to ask me or the TAs for help.³



Thomas Liu



Siddharth Sagar



Ozlem Yildiz

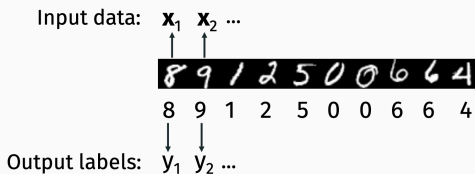
³Fill out office hours poll on Ed!

Class participation accounts for 10% of your grade. It's easy to get a perfect score:

- Ask and answer questions in lecture.
- Post questions or responses to other students on Ed. Or other things you find interesting.
- Participate in professor or TA office hours.

SIMPLE LINEAR REGRESSION

Step 1: Collect and label many input/output pairs (\mathbf{x}_i, y_i) . For our digit images, we have each $\mathbf{x}_i \in \mathbb{R}^{28 \times 28}$ and $y_i \in \{0, 1, \dots, 9\}$.



This is called the **training dataset**.

Step 2: Learn from the examples we have.

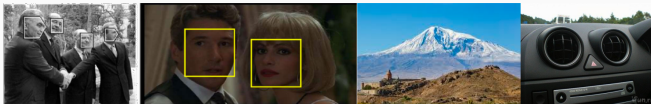
- Have the computer automatically find some function $f(\mathbf{x})$ such that $f(\mathbf{x}_i) \approx y_i$ for most (\mathbf{x}_i, y_i) in our training data set (by searching over many possible functions).

In **supervised learning** every input \mathbf{x}_i in our training dataset comes with a desired output y_i (typically generated by a human, or some other process).

Types of supervised learning:

- **Classification** – predict a discrete class label.
- **Regression** – predict a continuous value.
 - Dependent variable, response variable, target variable, lots of different names for y_i .

Another example of supervised classification: **Face Detection**.



Each input data example x_i is an image. Each output y_i is 1 if the image contains a face, 0 otherwise.

- Harder than digit recognition, but we now have very reliable methods (used in nearly all digital cameras, phones, etc.)

Other examples of supervised classification:

- Object detection (Input: image, Output: dog or cat)
- Spam detection (Input: email text, Output: spam or not)
- Medical diagnosis (Input: patient data, Output: disease condition or not)
- Credit decision making (Input: financial data, Output: offer loan or not)

Example of supervised regression: **Stock Price Prediction.**



Each input x is a vector of metrics about a company (sales volume, PE ratio, earning reports, historical price data).

Each output y_i is the **price of the stock** 3 months in the future.

Other examples of supervised regression:

- Home price prediction (Inputs: square footage, zip code, number of bathrooms, Output: Price)
- Car price prediction (Inputs: make, model, year, miles driven, Output: Price)
- Weather prediction (Inputs: weather data at nearby stations, Output: tomorrows temperature)
- Robotics/Control (Inputs: information about environment and current position at time t , Output: estimate of position at time $t + 1$)

Later in the class we will talk about other models:

- **Unsupervised learning** (no labels or response variable)
 - Clustering
 - Representation Learning
- **Reinforcement learning**
 - Game playing

You might also hear about semi-supervised learning or active learning – these categories aren't always cut and dry.

In **supervised learnings** every input x_i in our training dataset comes with a desired output y_i (typically generated by a human, or some other process).

Types of supervised learning:

- **Classification** – predict a discrete class label.
- **Regression** – predict a continuous value.
 - Dependent variable, response variable, target variable, lots of different names for y_j .

Motivating example: Predict the highway miles per gallon (MPG) of a car given quantitative information about its engine.
Demo in `demo_auto_mpg.ipynb`.

What factors might matter?

Data set available from the UCI Machine Learning Repository:
<https://archive.ics.uci.edu/>.

The screenshot shows the UCI Machine Learning Repository homepage. At the top, there is a navigation bar with links for 'About', 'Citation Policy', 'Donate a Data Set', and 'Contact'. Below the navigation bar is a search bar and a 'View ALL Data Sets' link. The main content area features a welcome message and a list of 'Latest News' on the left, 'Newest Data Sets' in the center, and 'Most Popular Data Sets (since 2007)' on the right. The 'Latest News' section includes updates from 2018 and 2019. The 'Newest Data Sets' section lists 15 recent additions with their dates and titles. The 'Most Popular Data Sets' section lists 10 frequently accessed datasets.

UCI Machine Learning Repository
 Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Repository Web Search

View ALL Data Sets

Welcome to the UC Irvine Machine Learning Repository!

We currently maintain 485 data sets as a service to the machine learning community. You may [view all data sets](#) through our searchable interface. For a general overview of the Repository, please visit our [About page](#). For information about citing data sets in publications, please read our [citation policy](#). If you wish to donate a data set, please consult our [donation policy](#). For any other questions, feel free to [contact the Repository librarians](#).

Supported By: In Collaboration With: **REXA** Info

Latest News:

- 06-24-2018: Welcome to the new Repository admins Dheeru Dua and El Karra Terrikidoul
- 04-04-2018: Welcome to the new Repository admins Kevin Bache and Moshe Lohman
- 03-07-2018: Info from donor regarding Kettle data
- 10-16-2009: Two new data sets have been added.
- 09-14-2009: Several data sets have been added.
- 03-04-2008: New data sets have been added!
- 06-25-2007: Two new data sets have been added: UJI Pen Characters, MAGIC Gamma Telescope

Featured Data Set: Ozone Level Detection

 **Task:** Classification
Data Type: Multivariate, Sequential, Time-Series
Attributes: 73
Instances: 2536

Two ground ozone level data sets are included in this collection. One is the eight hour peak set (ep8hr.data), the other is the one hour peak set (onehr.data). Those data were collected from 1988 to 2004 at the Houston, Galveston and Brazoria area.

Newest Data Sets:

- 10-06-2019: [WISDM Smartphones and Smartwatch Activity and Biometrics Dataset](#)
- 09-30-2019: [Heracleia C.Virus \(HCV\) for Egyptian patients](#)
- 09-23-2019: [CGAR fish toxicity](#)
- 09-23-2019: [CGAR aquatic toxicity](#)
- 09-21-2019: [Online Retail II](#)
- 09-20-2019: [Human Activity Recognition from Continuous Ambient Sensor Data](#)
- 09-20-2019: [Beijing Multi-Site Air-Quality Data](#)
- 09-20-2019: [MEs](#)
- 07-30-2019: [PPD-DatLA](#)
- 07-04-2019: [Diabetes Predictors data set](#)
- 07-22-2019: [Alcohol QCM Sensor Dataset](#)
- 07-14-2019: [Incident management process enriched event log](#)

Most Popular Data Sets (since 2007):

- 3099461: [Iris](#)
- 1711996: [Adult](#)
- 1326924: [Wine](#)
- 1126497: [Heart Disease](#)
- 1120880: [Wine Quality](#)
- 1116403: [Car Evaluation](#)
- 1110592: [Breast Cancer Wisconsin \(Diagnostic\)](#)
- 1101179: [Bank Marketing](#)
- 895298: [Human Activity Recognition Using Smartphones](#)
- 869144: [Abalone](#)
- 839187: [Forest Fires](#)
- 666591: [Poker Hand](#)

PREDICTING MPG

Datasets from UCI (and many other places) comes as tab, space, or comma delimited files.

Row	displacement	horsepower	weight	acceleration	mpg	car name			
1	18.0	8	307.0	130.0	3504.	12.0	70	1	<u>"chevrolet chevelle malibu"</u>
2	15.0	8	350.0	165.0	3693.	11.5	70	1	<u>"buick skylark 320"</u>
3	18.0	8	318.0	150.0	3436.	11.0	70	1	<u>"plymouth satellite"</u>
4	16.0	8	304.0	150.0	3433.	12.0	70	1	<u>"amc rebel sst"</u>
5	17.0	8	302.0	140.0	3449.	10.5	70	1	<u>"ford torino"</u>
6	15.0	8	429.0	198.0	4341.	10.0	70	1	<u>"ford galaxie 500"</u>
7	14.0	8	454.0	220.0	4354.	9.0	70	1	<u>"chevrolet impala"</u>
8	14.0	8	440.0	215.0	4312.	8.5	70	1	<u>"plymouth fury iii"</u>
9	14.0	8	455.0	225.0	4425.	10.0	70	1	<u>"pontiac catalina"</u>
10	15.0	8	390.0	190.0	3850.	8.5	70	1	<u>"amc ambassador dpl"</u>
11	15.0	8	383.0	170.0	3563.	10.0	70	1	<u>"dodge challenger se"</u>
12	14.0	8	340.0	160.0	3609.	8.0	70	1	<u>"plymouth 'cuda 340"</u>
13	15.0	8	400.0	150.0	3761.	9.5	70	1	<u>"chevrolet monte carlo"</u>
14	14.0	8	455.0	225.0	3086.	10.0	70	1	<u>"buick estate wagon (sw)"</u>
15	24.0	4	113.0	95.00	2372.	15.0	70	3	<u>"toyota corona mark ii"</u>
16	22.0	6	198.0	95.00	2833.	15.5	70	1	<u>"plymouth duster"</u>
17	18.0	6	199.0	97.00	2774.	15.5	70	1	<u>"amc hornet"</u>
18	21.0	6	200.0	85.00	2587.	16.0	70	1	<u>"ford maverick"</u>
19	27.0	4	97.00	88.00	2130.	14.5	70	3	<u>"datsun pl510"</u>
20	26.0	4	97.00	46.00	1835.	20.5	70	2	<u>"volkswagen 1131 deluxe sedan"</u>
21	25.0	4	110.0	87.00	2672.	17.5	70	2	<u>"peugeot 504"</u>
22	24.0	4	107.0	90.00	2430.	14.5	70	2	<u>"audi 100 ls"</u>
23	25.0	4	104.0	95.00	2375.	17.5	70	2	<u>"saab 99e"</u>
24	26.0	4	121.0	113.0	2234.	12.5	70	2	<u>"bmw 2002"</u>
25	21.0	6	199.0	90.00	2648.	15.0	70	1	<u>"amc gremlin"</u>
26	10.0	8	360.0	215.0	4615.	14.0	70	1	<u>"ford f250"</u>

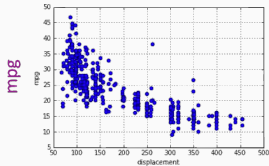
Check dataset description to know what each column means.

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
1	18.0	8	307.0	130.0	3504.	12.0	70	1	"chevrolet chevelle malibu"
2	15.0	8	350.0	165.0	3693.	11.5	70	1	"buick skylark 320"
3	18.0	8	318.0	150.0	3436.	11.0	70	1	"plymouth satellite"
4	16.0	8	304.0	150.0	3433.	12.0	70	1	"amc rebel sst"
5	17.0	8	302.0	140.0	3449.	10.5	70	1	"ford torino"
6	15.0	8	429.0	198.0	4341.	10.0	70	1	"ford galaxie 500"
7	14.0	8	454.0	220.0	4354.	9.0	70	1	"chevrolet impala"
8	14.0	8	440.0	215.0	4312.	8.5	70	1	"plymouth fury iii"
9	14.0	8	455.0	225.0	4425.	10.0	70	1	"pontiac catalina"
10	15.0	8	390.0	190.0	3850.	8.5	70	1	"amc ambassador dpl"
11	15.0	8	383.0	170.0	3563.	10.0	70	1	"dodge challenger se"
12	14.0	8	340.0	160.0	3609.	8.0	70	1	"plymouth 'cuda 340"
13	15.0	8	400.0	150.0	3761.	9.5	70	1	"chevrolet monte carlo"
14	14.0	8	455.0	225.0	3086.	10.0	70	1	"buick estate wagon (sw)"
15	24.0	4	113.0	95.00	2372.	15.0	70	3	"toyota corona mark ii"
16	22.0	6	198.0	95.00	2833.	15.5	70	1	"plymouth duster"
17	18.0	6	199.0	97.00	2774.	15.5	70	1	"amc hornet"
18	21.0	6	200.0	85.00	2587.	16.0	70	1	"ford maverick"
19	27.0	4	97.00	88.00	2130.	14.5	70	3	"datsun pl510"
20	26.0	4	97.00	46.00	1835.	20.5	70	2	"volkswagen 1131 deluxe sedan"
21	25.0	4	110.0	87.00	2672.	17.5	70	2	"peugeot 504"
22	24.0	4	107.0	90.00	2430.	14.5	70	2	"audi 100 ls"
23	25.0	4	104.0	95.00	2375.	17.5	70	2	"saab 99e"
24	26.0	4	121.0	113.0	2234.	12.5	70	2	"bmw 2002"
25	21.0	6	199.0	90.00	2648.	15.0	70	1	"amc gremlin"
26	10.0	8	360.0	215.0	4615.	14.0	70	1	"ford f250"

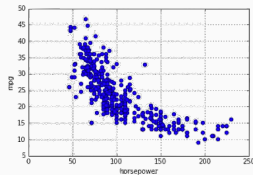
'mpg', 'cylinders', 'displacement', 'horsepower', 'weight',
'acceleration', 'model year', 'origin', 'car name'

LIBRARIES FOR INITIAL DATA READING

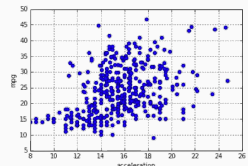
- Use `pandas` for reading data from delimited files. Stores data in a type of table called a “data frame” but this is just a wrapper around a `numpy` array.
- Use `matplotlib` for initial exploration.



Displacement



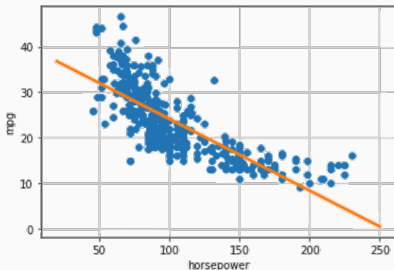
Horsepower



Acceleration

SIMPLE LINEAR REGRESSION

Linear regression from a Machine Learning (not a Statistics) perspective. Our first supervised machine learning model.

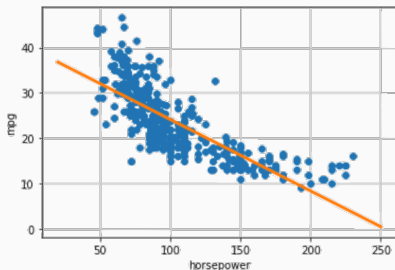


Only focus on one predictive variable at a time (e.g. horsepower). This is why it's called simple linear regression.

SIMPLE LINEAR REGRESSION

Dataset:

- $x_1, \dots, x_n \in \mathbb{R}$ (horsepowers of n cars – this is the predictor/independent variable)
- $y_1, \dots, y_n \in \mathbb{R}$ (MPG – this is the response/dependent variable)



SUPERVISED LEARNING DEFINITIONS

- **Model** $f_{\theta}(x)$: Class of equations or programs which map input x to predicted output. We want $f_{\theta}(x_i) \approx y_i$ for training inputs.
- **Model Parameters** θ : Vector of numbers. These are numerical knobs which parameterize our class of models.
- **Loss Function** $L(\theta)$: Measure of how well a model fits our data. Often some function of $f_{\theta}(x_1) - y_1, \dots, f_{\theta}(x_n) - y_n$

Common Goal: Choose parameters θ^* which minimize the Loss Function:

$$\theta^* = \arg \min_{\theta} L(\theta)$$

Choosing θ^* based on minimizing the empirical error on our training data is called Empirical Risk Minimization. It is by far the most common approach to solving supervised learning problems.

General Supervised Learning

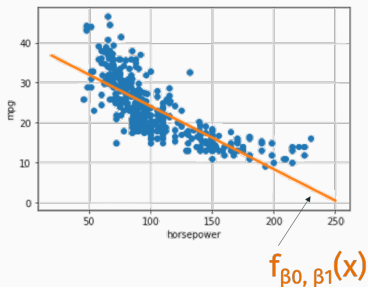
- Model: $f_{\theta}(x)$
- Model Parameters: θ
- Loss Function: $L(\theta)$

Linear Regression

- Model:
- Model Parameters:
- Loss Function:

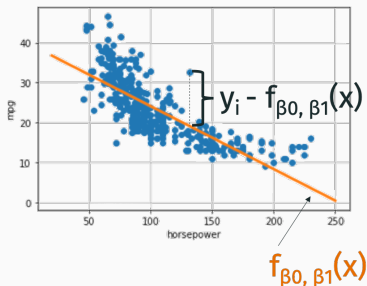
HOW TO MEASURE GOODNESS OF FIT

What is a natural **loss function** for linear regression?



HOW TO MEASURE GOODNESS OF FIT

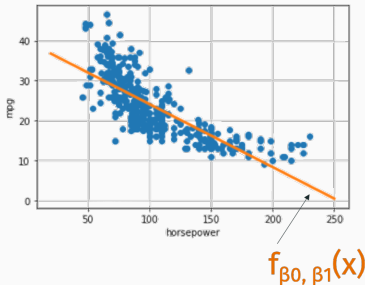
Typical choices are a function of $y_1 - f_{\beta_0, \beta_1}(x_1), \dots, y_n - f_{\beta_0, \beta_1}(x_n)$



- l_2 /Squared Loss: $L(\beta_0, \beta_1) = \sum_{i=1}^n [y_i - f_{\beta_0, \beta_1}(x_i)]^2$.
- l_1 /Least absolute deviations: $L(\beta_0, \beta_1) = \sum_{i=1}^n |y_i - f_{\beta_0, \beta_1}(x_i)|$.
- l_∞ Loss $L(\beta_0, \beta_1) = \max_{i \in \{1, \dots, n\}} |y_i - f_{\beta_0, \beta_1}(x_i)|$.

HOW TO MEASURE GOODNESS OF FIT

We're going to start with the Squared Loss/Sum-of-Squares Loss. Also called "Residual Sum-of-Squares (RSS)"



- Relatively robust to outliers.
- Simple to define, leads to simple algorithms for finding β_0, β_1
- Theoretically justified from classical statistics related to assumptions about Gaussian noise. Will discuss later in the course.

General Supervised Learning

- Model: $f_{\theta}(x)$
- Model Parameters: θ
- Loss Function: $L(\theta)$

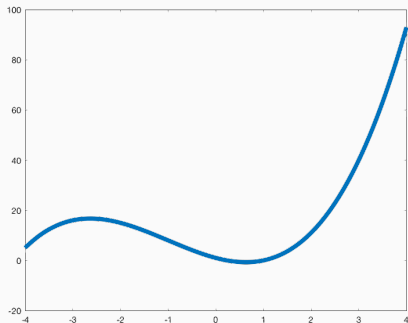
Linear Regression

- Model:
 $f_{\beta_0, \beta_1}(x) = \beta_0 + \beta_1 \cdot x$
- Model Parameters: β_0, β_1
- Loss Function: $L(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - f_{\beta_0, \beta_1}(x_i))^2$

Goal: Choose β_0, β_1 to minimize
 $L(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$.

This is the entire job of any **Supervised Learning Algorithm**.

Univariate function:



$$x^3 + 3 \cdot x^2 - 5 \cdot x + 1$$

- Find all places where derivative $f'(x) = 0$ and check which has the smallest value.

Multivariate function: $L(\beta_0, \beta_1)$

- Find values of β_0, β_1 where all partial derivatives equal 0.
- $\frac{\partial L}{\partial \beta_0} = 0$ and $\frac{\partial L}{\partial \beta_1} = 0$.

Multivariate function: $L(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$

- Find values of β_0, β_1 where all partial derivatives equal 0.
- $\frac{\partial L}{\partial \beta_0} = 0$ and $\frac{\partial L}{\partial \beta_1} = 0$.

Some definitions:

- Let $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$. \bar{y} is the mean of y .
- Let $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$. \bar{x} is the mean of x .
- Let $\sigma_y^2 = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2$. σ_y^2 is the variance of y .
- Let $\sigma_x^2 = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2$. σ_x^2 is the variance of x .
- Let $\sigma_{xy} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$. σ_{xy} is the covariance.

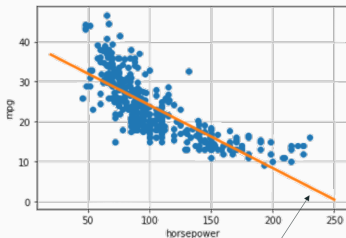
Claim: $L(\beta_0, \beta_1)$ is minimized when:

- $\beta_1 = \sigma_{xy} / \sigma_x^2$
- $\beta_0 = \bar{y} - \beta_1 \bar{x}$

MINIMIZING SQUARED LOSS FOR REGRESSION

Takeaways:

- Minimizing functions is sometimes easy with calculus, but not often! We will learn much more general tools (like gradient descent).
- Simple closed form formula for optimal parameters β_0^* and β_1^* for squared-loss!



$$B_0^* + B_1^*x$$

Let $L(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$.

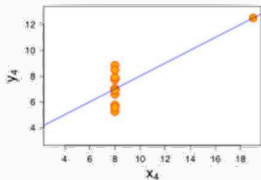
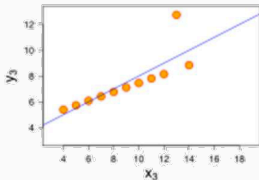
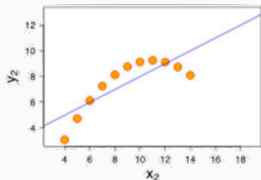
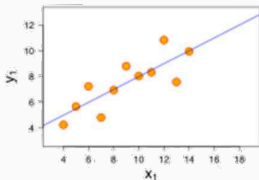
$$R^2 = 1 - \frac{L(\beta_0, \beta_1)}{n\sigma_y^2}$$

is exactly the R^2 value (“coefficient of determination”) you may remember from statistics.

The smaller the loss, the closer R^2 is to 1, which means we have a better regression fit.

A FEW COMMENTS

Many reasons you might get a poor regression fit:

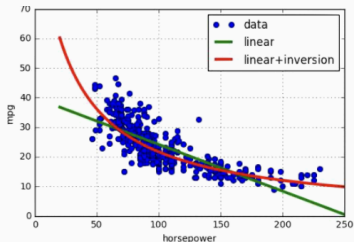
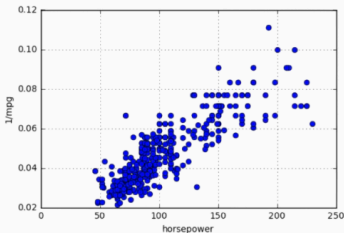


A FEW COMMENTS

Some of these are fixable!

- Remove outliers, use more robust loss function.
- **Non-linear model transformation.**

Fit the model $\frac{1}{\text{mpg}} \approx \beta_0 + \beta_1 \cdot \text{horsepower}$.



Much better fit, same exact learning algorithm!