WELCOME

Introduction to Machine Learning

DBDA.X408.(34)

Instructor:

Bill Chen

UCSC Silicon Valley Extension
PROFESSIONAL EDUCATION

UCSC Silicon Valley Extension

Email: xchen375@ucsc.edu

Bill's Background: Machine/Deep learning Engineer

Current research:

- Surgical CV analysis: A self-supervised learning approach is proposed to utilize robotic surgery videos for automating two critical OR tasks: detecting anomalies and estimating remaining surgery time, with promising results in improving patient safety, comfort, and resource optimization by streamlining OR tasks.

Current job:

- Develop machine learning tools to generate synthetic data for Language Model.

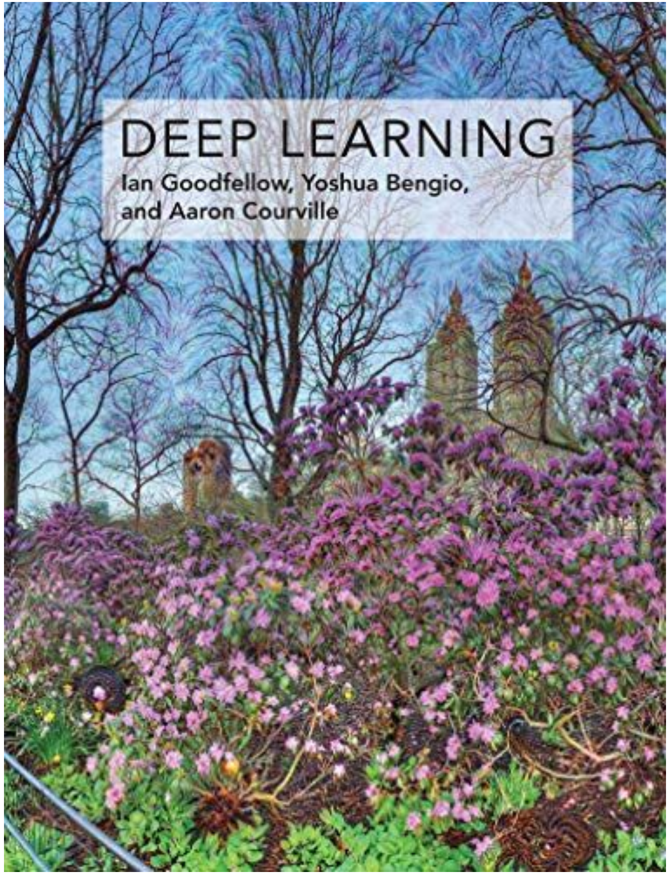
Learning Goals

- Identify and formulate ML problems
- Understand and implement algorithms to solve ML problems
- Explain the implementation, working, and practical benefit of many ML topics
- Analyze the performance of given or implemented ML solutions on practical datasets.

Textbook

Optional:

- Deep Learning (free online): https://www.deeplearningbook.org/
- Very comprehensive, more like a lookup reference



Performance Evaluation

- **Quizzes (30%):** There are 9 quizzes total, but only top 6 highest grades are included in the final grade.
- **Problem sets (30%):** There are 4 problem sets that requires student hands-on working on what they have learnt for the past two weeks. Top 3 grades will be counted into the final grade.
- **Take-home exam (20%):** A final take-home exam covering the entire course content will be assigned after the last class.
- **Final project (20%):** The project must include these steps: data collection, data preparation, classifier design, and performance evaluation.
  - Open-ended, teams of 1-3.

Honor Code, and Questions?

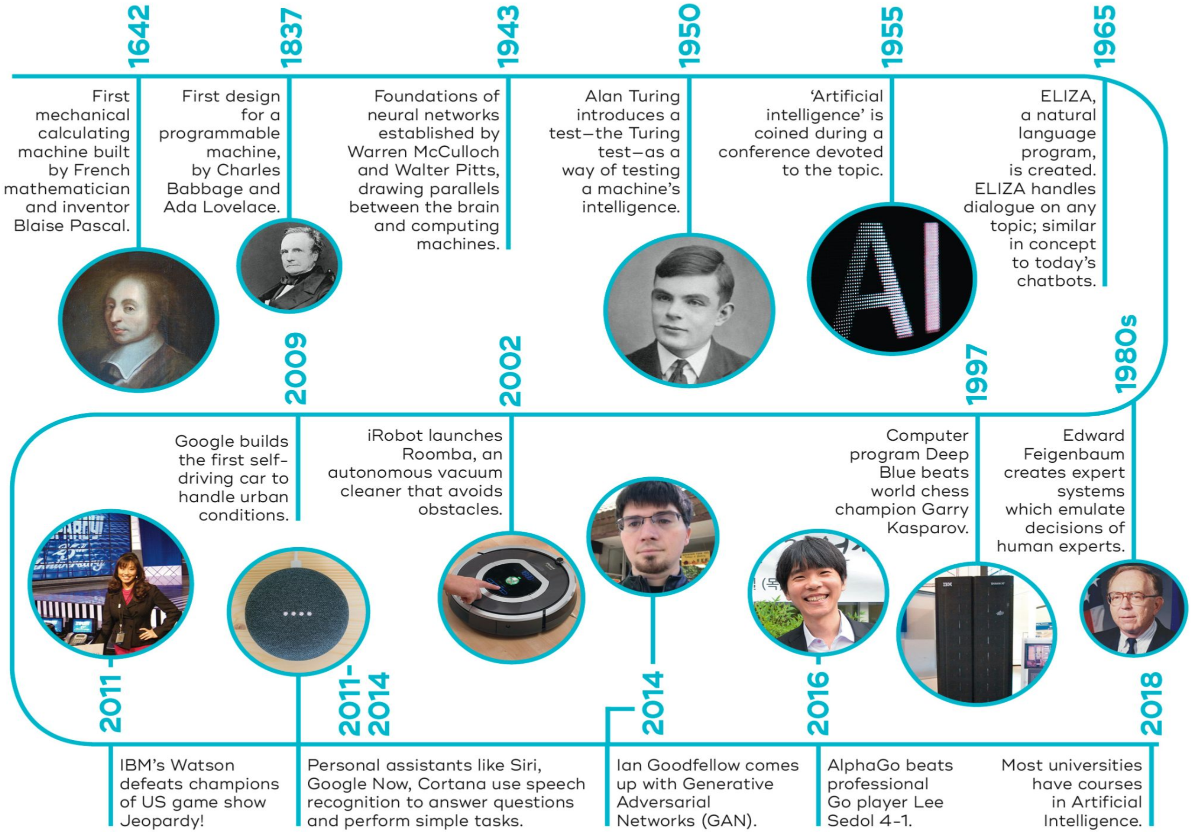
Do’s

- form study groups (with arbitrary number of people); discuss and work on homework problems in groups
- write down the solutions independently
- write down the names of people with whom you’ve discussed the homework

Don’ts

- It is an honor code violation to copy, refer to, or look at written or code solutions from a previous year, including but not limited to: official solutions from a previous year, solutions posted online, solutions you or someone else may have written up in a previous year, and solutions for related problems.

Timeline of Artificial Intelligence



Definition of Machine Learning

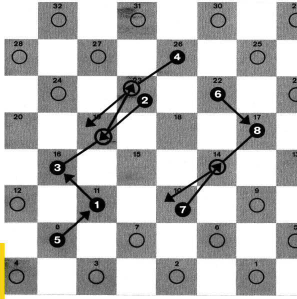
Arthur Samuel (1959): Machine Learning is the
field of study that gives the computer the ability
to learn without being explicitly programmed.

Examples are used to train computers to perform tasks that would be **difficult to program**.

A. L. Samuel

Some Studies in Machine Learning Using the Game of Checkers

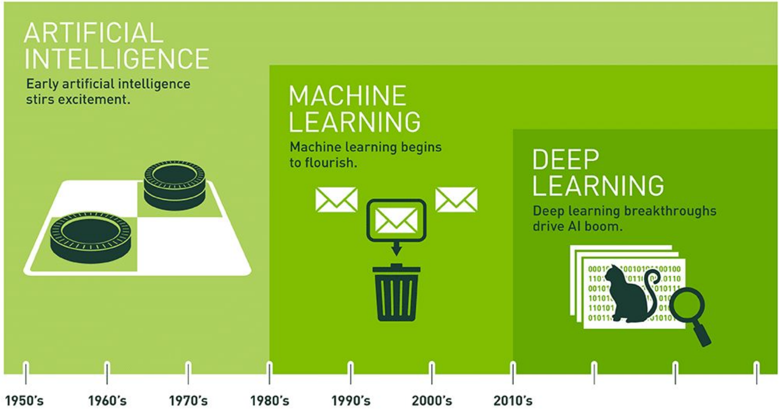
Abstract: Two machine-learning procedures have been investigated in some detail using the game of checkers. Enough work has been done to verify the fact that a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program. Furthermore, it can learn to do this in a remarkably short period of time (8 or 10 hours of machine-playing time) when given only the rules of the game, a sense of direction, and a redundant and incomplete list of parameters which are thought to have something to do with the game, but whose correct signs and relative weights are unknown and unspecified. The principles of machine learning verified by these experiments are, of course, applicable to many other situations.

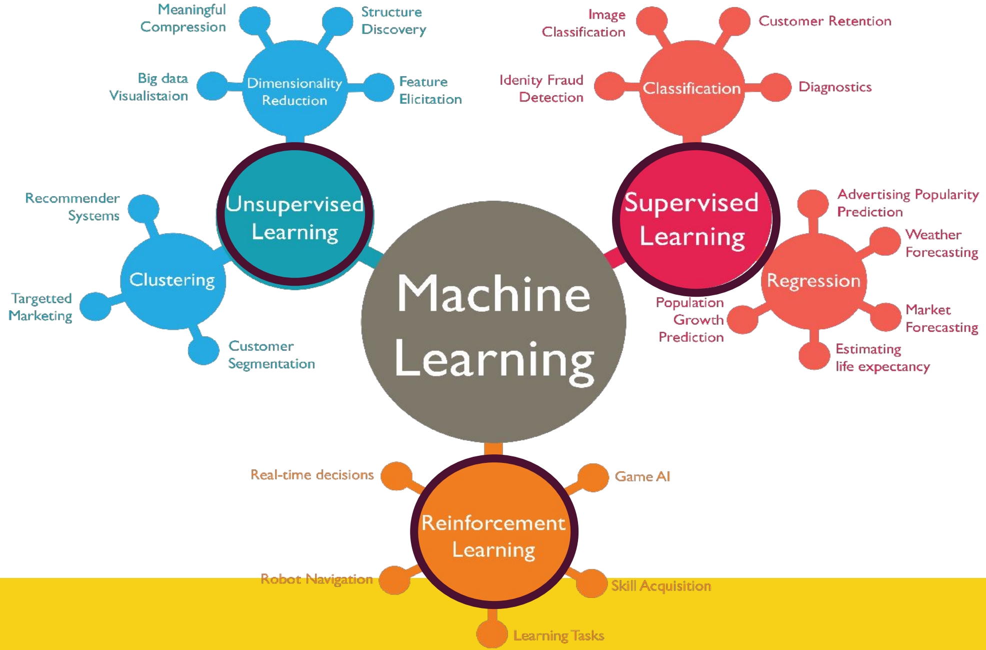


Eight-move opening utilizing generalization learning. (See Appendix B, Game G-43.)

Taxonomy of Machine Learning

- Supervised Learning
  - Training data is labeled
  - Goal is correctly label new data
- Reinforcement Learning
  - Training data is unlabeled
  - System receives feedback for its actions
  - Goal is to perform better actions
- Unsupervised Learning
  - Training data is unlabeled
  - Goal is to categorize the observations

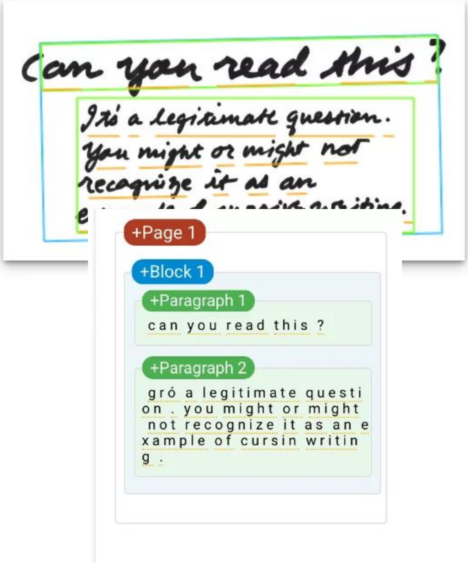


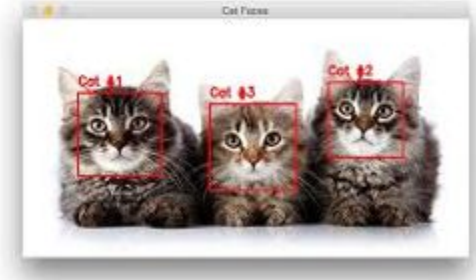


Supervised Learning

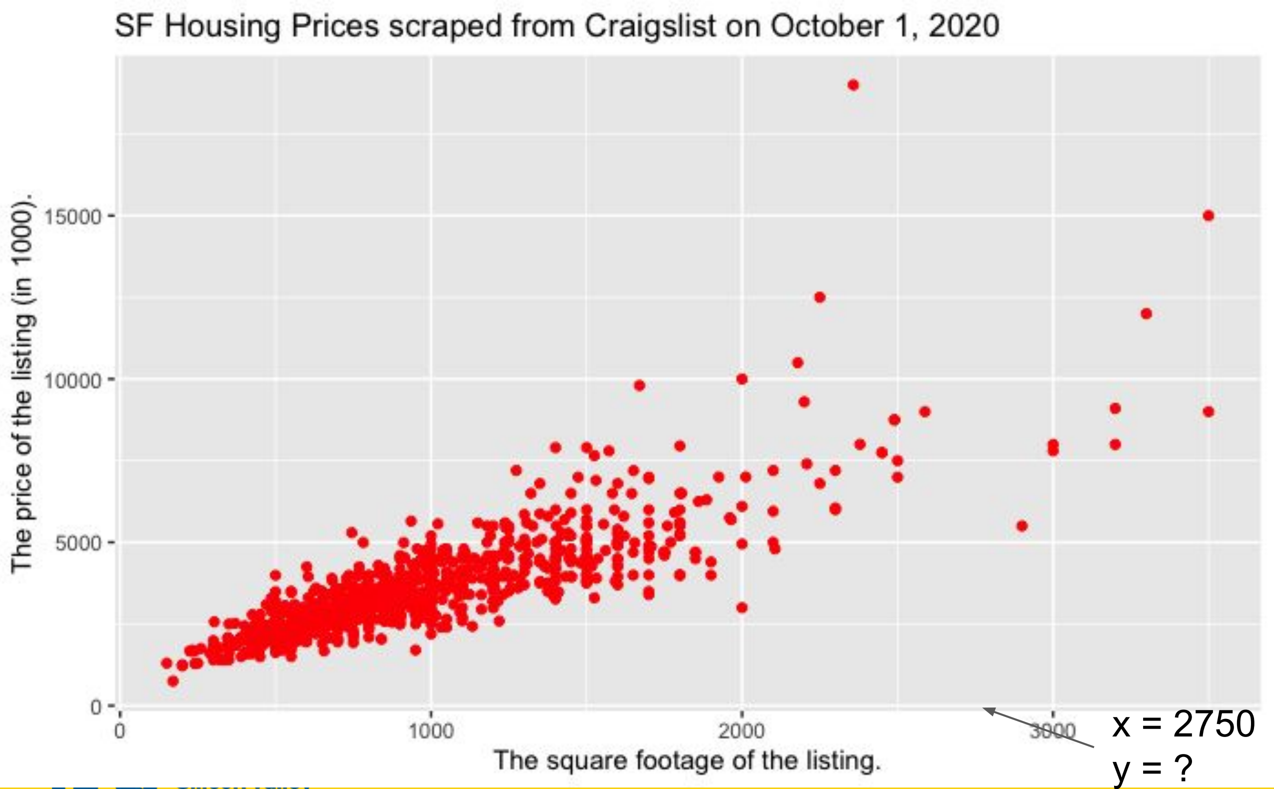
Applications of Machine Learning

- Handwriting Recognition
  - convert written letters into digital letters
- Language Translation
  - translate spoken and or written languages (e.g. Google Translate)
- Speech Recognition
  - convert voice snippets to text (e.g. Siri, Cortana, and Alexa)
- Image Classification
  - label images with appropriate categories (e.g. Google Photos)
- Autonomous Driving
  - enable cars to drive





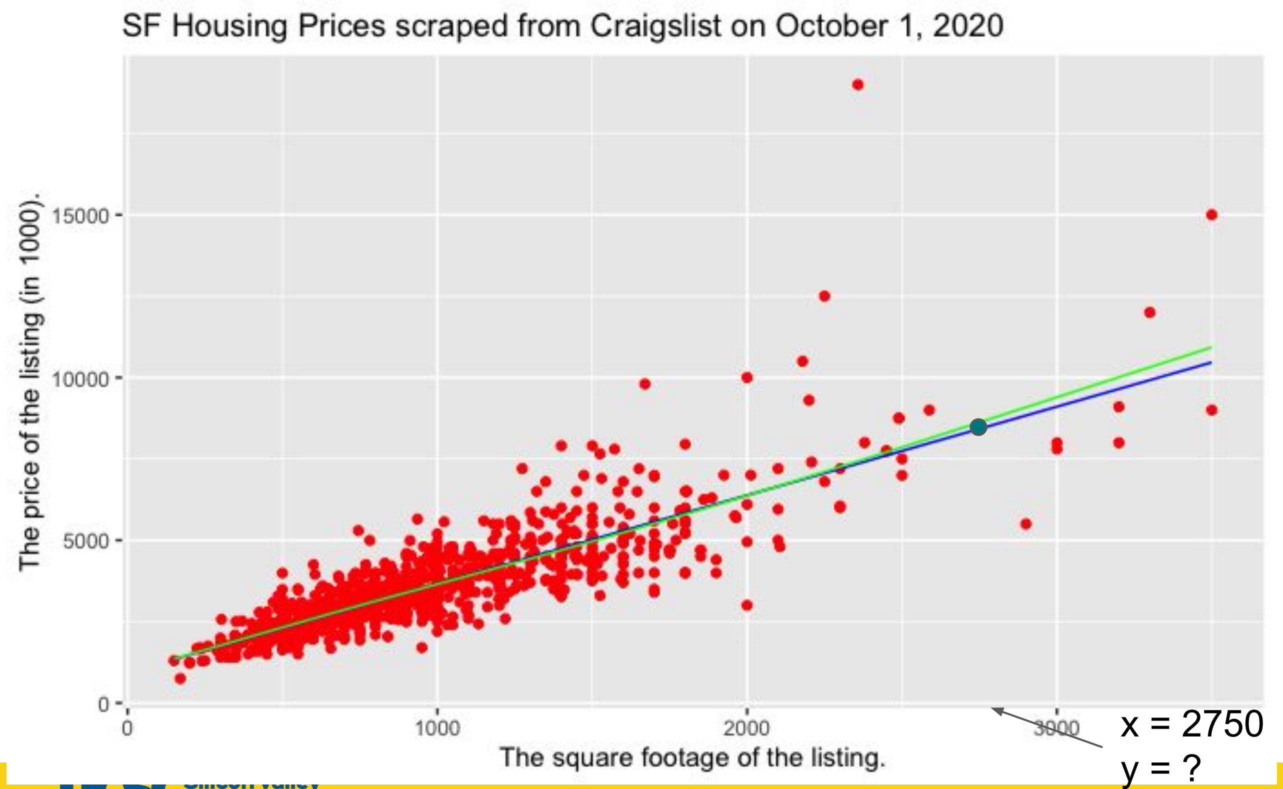
Example: Regression (Housing Prices Prediction)



Given: a dataset that contains n samples $(x^{(1)}, y^{(1)}), \dots (x^{(n)}, y^{(n)})$

Task: if a residence has x square feet, predict its price?

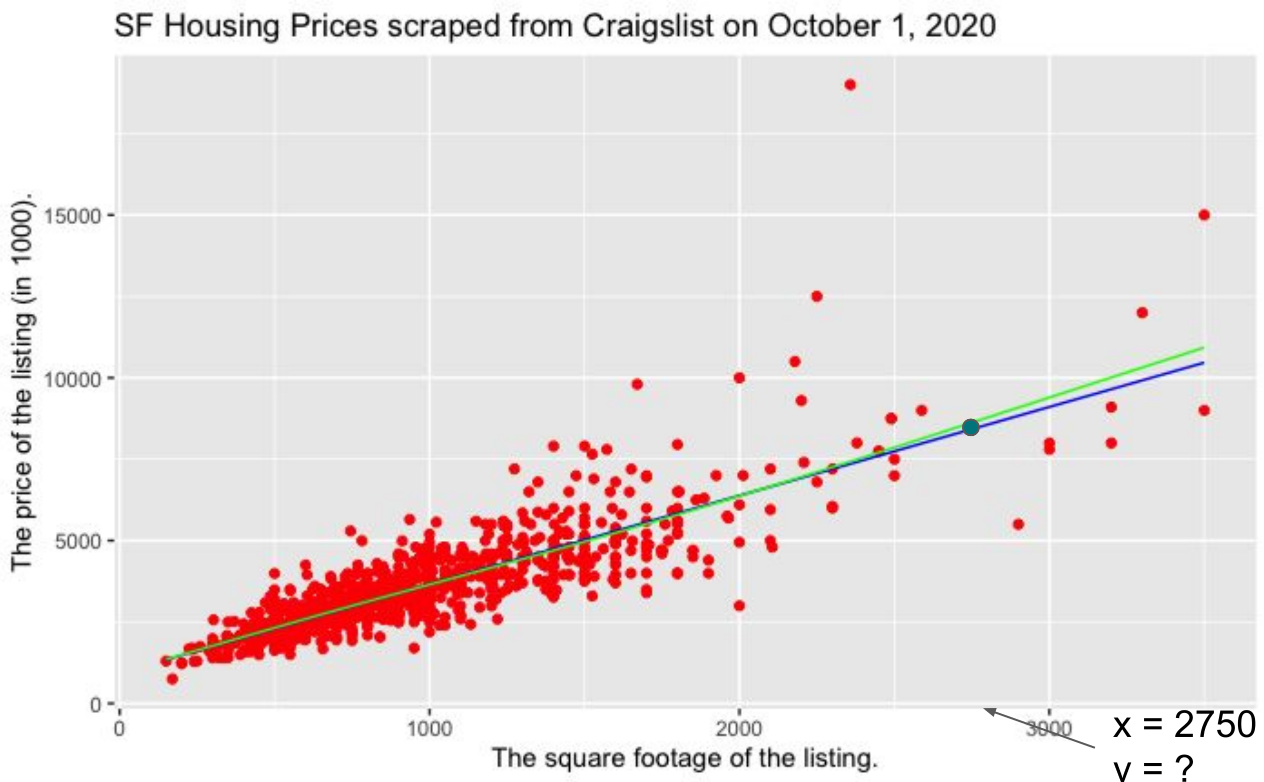
Example: Regression (Housing Prices Prediction)



Given: a dataset that contains n samples $(x^{(1)}, y^{(1)}), \ldots (x^{(n)}, y^{(n)})$

Task: if a residence has x square feet, predict its price?

Example: Regression (Housing Prices Prediction)



x = 2750
y = ?

Given: a dataset that contains n samples $(x^{(1)}, y^{(1)}), \ldots (x^{(n)}, y^{(n)})$

Task: if a residence has *x* square feet, predict its price?

We also know how many rooms in the house.

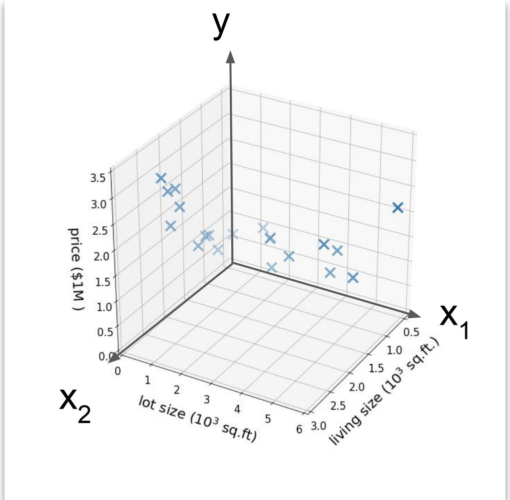
(size, room#) → price
features/input label/output
$x \in R^2$ $y \in R$

Dataset now for i-th element:

$$\boldsymbol{x}^{(i)} = \left(x_1^{(i)}, x_2^{(i)} \right)$$

Example: Regression (Housing Prices Prediction)

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_d \end{bmatrix} \begin{array}{l} \text{--- living size} \\ \text{--- lot size} \\ \text{--- \# floors} \\ \text{--- condition} \\ \text{--- zip code} \\ \qquad\vdots \\ \\ \end{array} \longrightarrow y \text{ --- price}$$



What values to use?

Are they good or bad?

- Size
- No. of rooms
- Parking
- School
- Crime
- Color
- No. of windows
- Door style

High-dimensional Features $x \in R^d$

Features in Machine Learning

- Features are the observations that are used to form predictions
  - For image classification, the pixels are the features
  - For voice recognition, the pitch and volume of the sound samples are the features
  - For autonomous cars, data from the cameras, range sensors, and GPS are features
- Extracting relevant features is important for building a model
  - Time of day is an irrelevant feature when classifying images
  - Time of day is relevant when classifying emails because SPAM often occurs at night
- Common Types of Features in Robotics
  - Pixels (RGB data)
  - Depth data (sonar, laser rangefinders)
  - Movement (encoder values)
  - Orientation or Acceleration (Gyroscope, Accelerometer, Compass)

Regression vs Classification

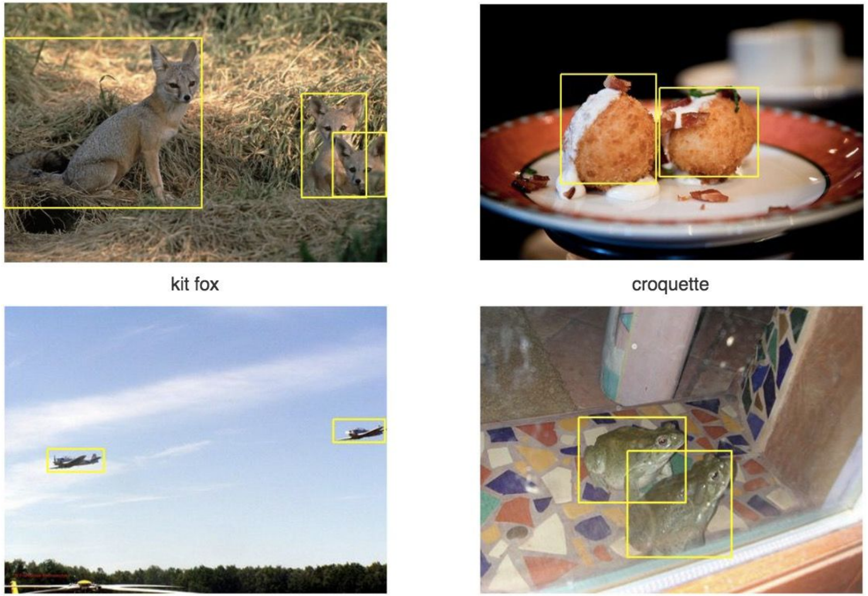
- Regression: if $y \in \mathbb{R}$ is a continuous variable
  - e.g., price prediction
- Classification: the label is a discrete variable
  - e.g., the task of predicting the types of residence

- Image Classification
  - x = raw pixels of the image,
  - y = the main object

ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Regression vs Classification

- Regression: if $y \in \mathbb{R}$ is a continuous variable
    - e.g., price prediction
- Classification: the label is a discrete variable
    - e.g., the task of predicting the types of residence
- Object localization and detection
    - x = raw pixels of the image,
    - y = the bounding boxes



kit fox croquette airplane frog

ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

Regression vs Classification

- Regression: if $y \in \mathbb{R}$ is a continuous variable
  - e.g., price prediction
- Classification: the label is a discrete variable
  - e.g., the task of predicting the types of residence

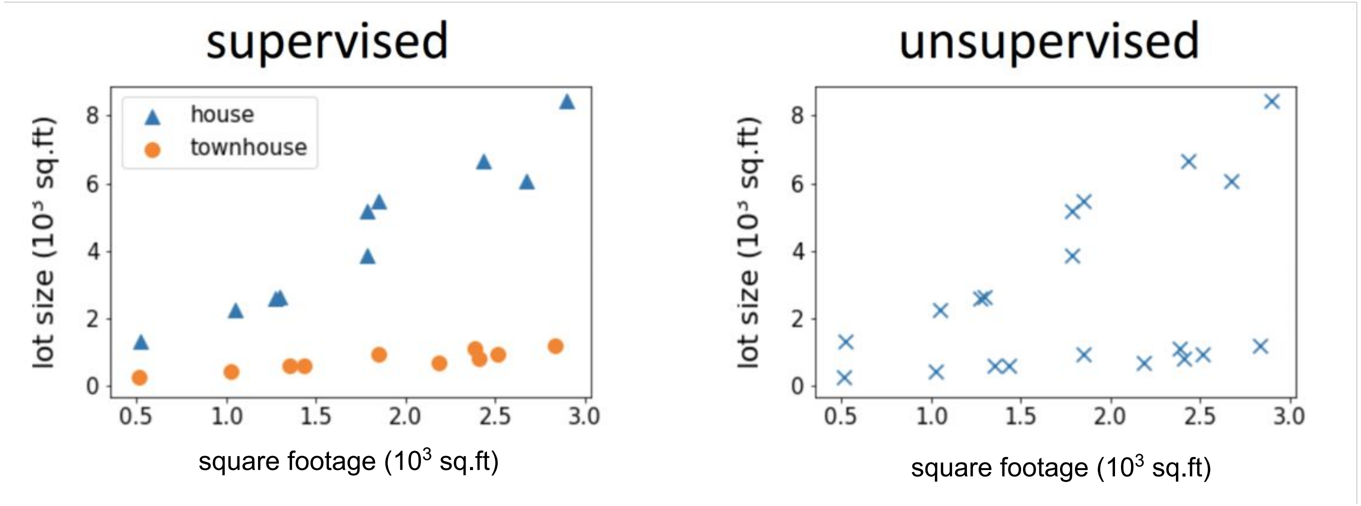
- Machine translation
  - x = text/token,
  - y = token/text
-

ImageNet Large Scale Visual Recognition Challenge. Russakovsky et al.'2015

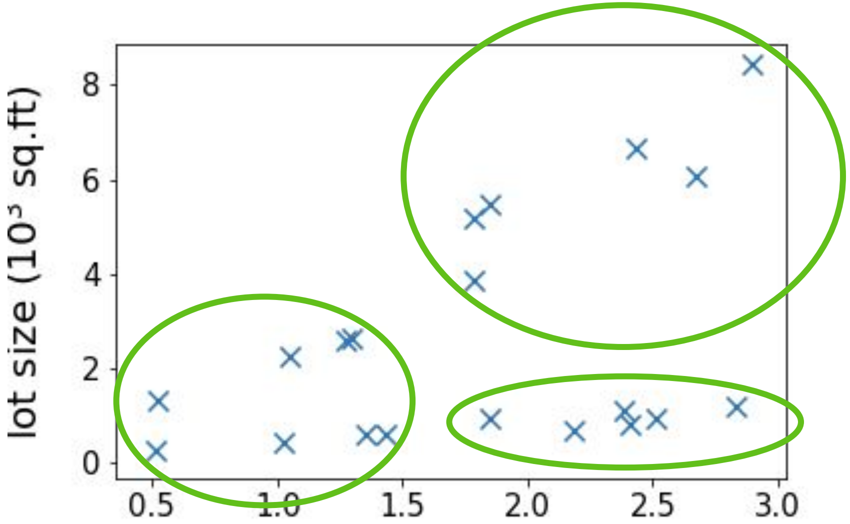
Unsupervised Learning

Unsupervised Learning

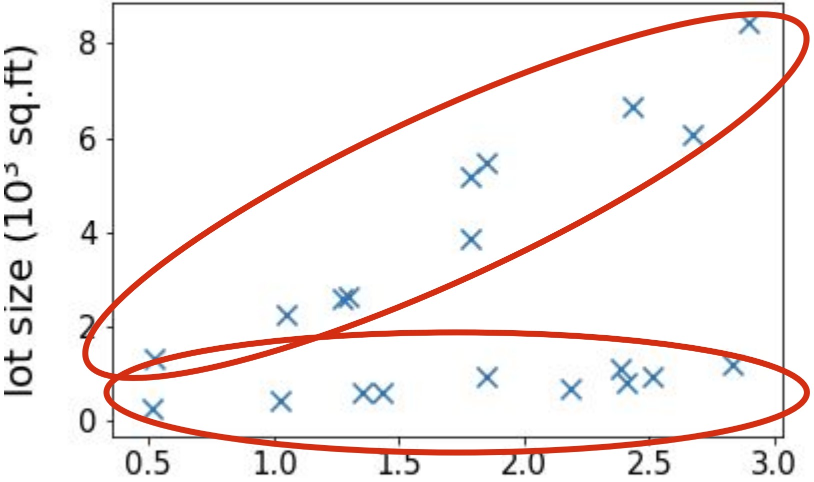
- Dataset contains no labels, aka no y.
- Goal: to understand the data and find meaningful structure in the data.



Clustering

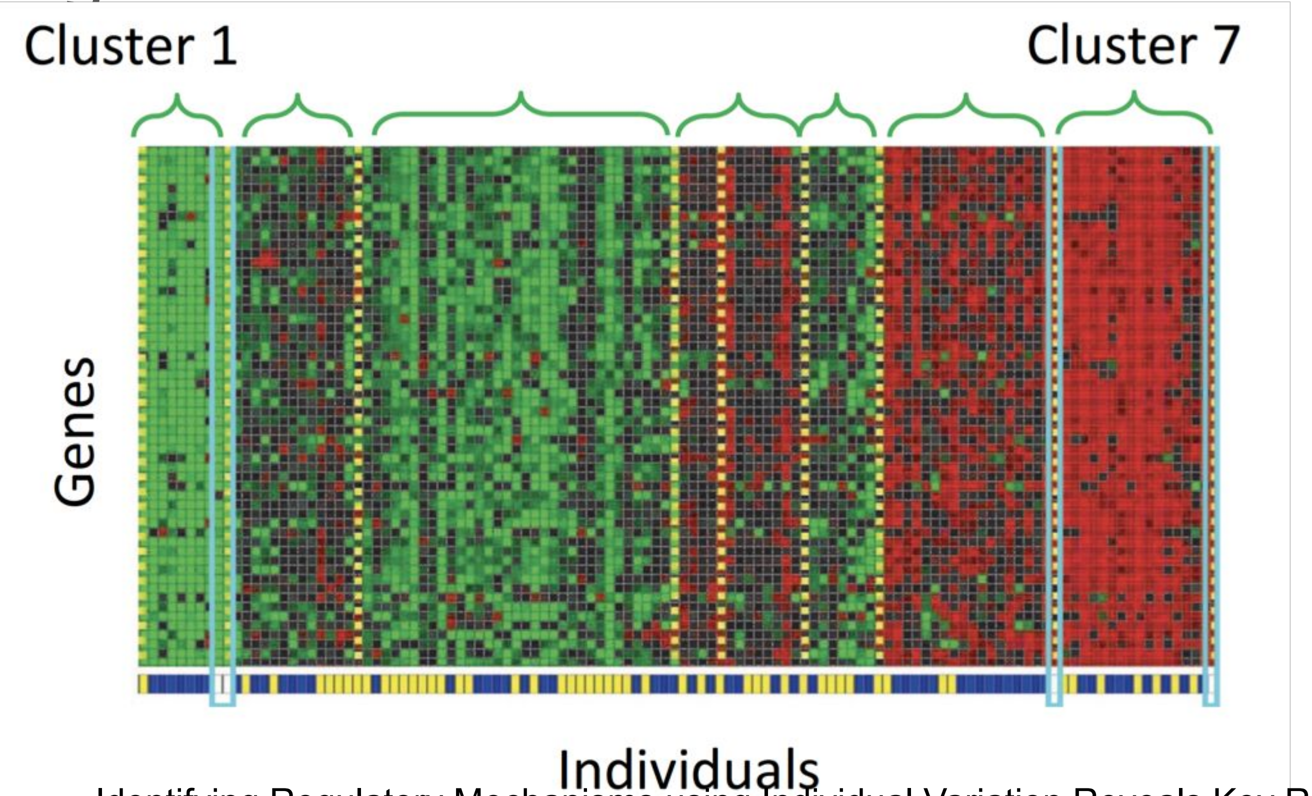


Clustering (k-mean clustering, mixture of Gaussians)



Note: it doesn't have to be circular shaped.

Clustering Genes



Identifying Regulatory Mechanisms using Individual Variation Reveals Key Role for Chromatin Modification. [Su-In Lee, Dana Pe'er, Aimee M. Dudley, George M. Church and Daphne Koller. '06]

TF-IDF

d_1 = "The cow jumped over the moon"

d_2 = "O'Leary's cow kicked the lamp"

d_3 = "The kicked lamp started a fire"

d_4 = "The cow on fire"

Doc	the	cow	jumped	over	moon	O'Leary's	kicked	lamp	started	fire	on	a
d_1	2	1	1	1	1	0	0	0	0	0	0	0
d_2	1	1	0	0	0	1	1	1	0	0	0	0
d_3	1	0	0	0	0	0	1	1	1	1	0	1
d_4	1	1	0	0	0	0	0	0	0	1	1	0

$sim(d_1, d_4)$ vs. $sim(d_2, d_4)$ vs. $sim(d_3, d_4)$?

Words	d_1	d_2	d_3	d_4
the	0	0	0	0
cow	0.42	0.42	0	0.42
jumped	2	0	0	0
over	2	0	0	0
moon	2	0	0	0
O'Leary's	0	2	0	0
kicked	0	1	1	0
lamp	0	1	1	0
started	0	0	2	0
fire	0	0	1	1
on	0	0	0	1
a	0	0	2	0

After TF-IDF

Term Frequency–Inverse Document Frequency: TF-IDF

$$TF(t) = \left(\frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}\right)$$

$$IDF(t) = \log_e\left(\frac{\sum \#\ of\ documents}{\#\ of\ documents\ with\ term\ t\ in\ it}\right)$$

TF-IDF Value for each word would be= TF(Value)*IDF(Value)

TF-IDF

d_1 = "The cow jumped over the moon"

d_2 = "O'Leary's cow kicked the lamp"

d_3 = "The kicked lamp started a fire"

d_4 = "The cow on fire"

Doc	the	cow	jumped	over	moon	O'Leary's	kicked	lamp	started	fire	on	a
d_1	2	1	1	1	1	0	0	0	0	0	0	0
d_2	1	1	0	0	0	1	1	1	0	0	0	0
d_3	1	0	0	0	0	0	1	1	1	1	0	1
d_4	1	1	0	0	0	0	0	0	0	1	1	0

$sim(d_1, d_4)$ vs. $sim(d_2, d_4)$ vs. $sim(d_3, d_4)$?

Term Frequency–Inverse Document Frequency: TF-IDF

$$TF(t) = \left(\frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}\right)$$

$$IDF(t) = \log_e\left(\frac{\sum \#\ of\ documents}{\#\ of\ documents\ with\ term\ t\ in\ it}\right)$$

TF-IDF Value for each word would be= TF(Value)*IDF(Value)

Words	d_1	d_2	d_3	d_4
the	0	0	0	0
cow	0.42	0.42	0	0.42
jumped	2	0	0	0
over	2	0	0	0
moon	2	0	0	0
O'Leary's	0	2	0	0
kicked	0	1	1	0
lamp	0	1	1	0
started	0	0	2	0
fire	0	0	1	1
on	0	0	0	1
a	0	0	2	0

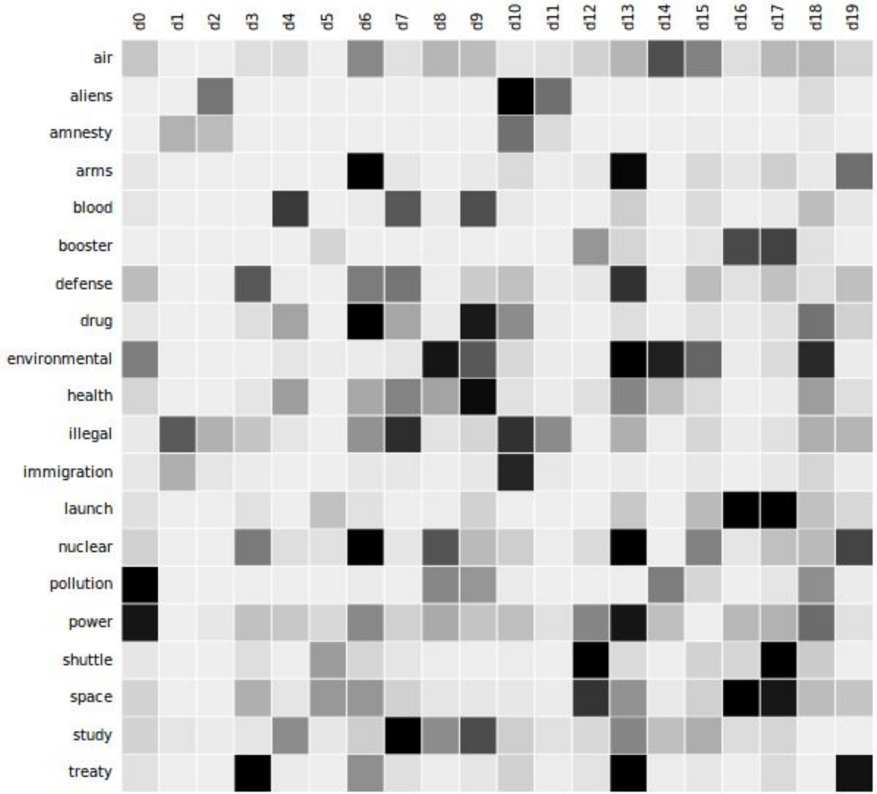
After TF-IDF

For example, to present a document d_5: "Oleary's cow on the moon":

For d_1: (0+0.4+2+0+0) / 5 = 0.48
For d_2: 0.48
For d_3: 0
For d_4: 0.284

So d_5={0.48, 0.48, 0.0, 0.284}

Latent Semantic Analysis (LSA)



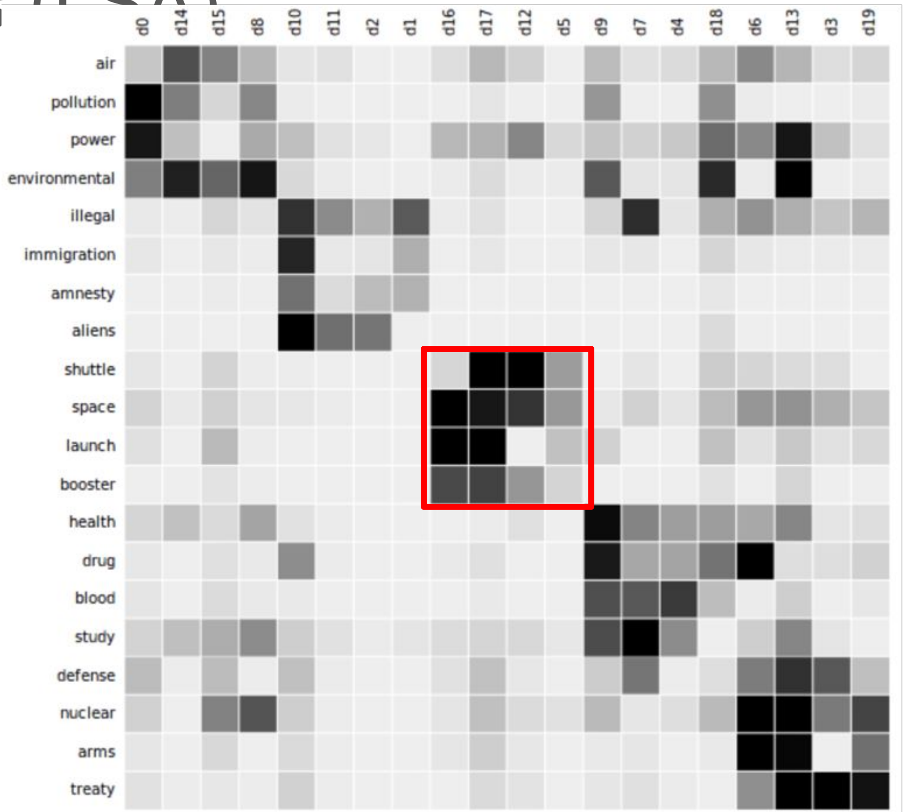
https://upload.wikimedia.org/wikipedia/commons/6/6e/Topic_detection_in_a_document-word_matrix.gif?20170329150506

Latent Semantic Analysis (LSA)

Singular decomposition analysis(SVD)

$$C_{m \times n} = U_{m \times r} \times \Sigma_{r \times r} \times V^{T}_{r \times n}$$

- C is the original word x document matrix with m words and n docs.
- U describes the original row entities as vectors of derived orthogonal factor values
- V describes the original column entities in the same way
- Σ is a diagonal matrix containing scaling values



https://upload.wikimedia.org/wikipedia/commons/6/6e/Topic_detection_in_a_document-word_matrix.gif?20170329150506

Latent Semantic Analysis (LSA)

Classic example from Deerwester 1990

Documents:

- c1: Human machine interface for ABC computer applications
- c2: A survey of user opinion of computer system response time
- c3: The EPS user interface management system
- c4: System and human system engineering testing of EPS
- c5: Relation of user perceived response time to error measurement
- m1: The generation of random, binary, ordered trees
- m2: The intersection graph of paths in trees
- m3: Graph minors IV: Widths

words	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Similarity between human and user is 0, if we use the co-occur of words in the document.

Solution:
LSA: a fully automatic mathematical/statistical technique for extracting and inferring relations between expected contextual usage of words in passages of discourse.

Singular decomposition analysis(SVD)

$$C_{m \times n} = U_{m \times r} \times \Sigma_{r \times r} \times V^{T}_{r \times n}$$

- C is the original word x document matrix with m words and n docs.
- U describes the original row entities as vectors of derived orthogonal factor values
- V describes the original column entities in the same way
- Σ is a diagonal matrix containing scaling values

Latent Semantic Analysis (LSA)

words	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

$$U = \begin{pmatrix}
-0.22 & -0.11 & 0.29 & -0.41 & -0.11 & -0.34 & -0.52 & 0.06 & 0.41 & -0.08 & 0.32 & -0.06 \\
-0.2 & -0.07 & 0.14 & -0.55 & 0.28 & 0.5 & 0.07 & 0.01 & 0.11 & -0.03 & -0.46 & -0.28 \\
-0.24 & 0.04 & -0.16 & -0.59 & -0.11 & -0.25 & 0.3 & -0.06 & -0.49 & 0.11 & 0.13 & 0.34 \\
-0.4 & 0.06 & -0.34 & 0.1 & 0.33 & 0.38 & -0. & 0. & -0.01 & -0.15 & 0.65 & -0.11 \\
-0.64 & -0.17 & 0.36 & 0.33 & -0.16 & -0.21 & 0.17 & -0.03 & -0.27 & -0.11 & -0.13 & -0.34 \\
-0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & -0.28 & 0.02 & 0.05 & -0.57 & -0.43 & 0.31 \\
-0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & -0.28 & 0.02 & 0.05 & 0.73 & -0.21 & -0.2 \\
-0.3 & -0.14 & 0.33 & 0.19 & 0.11 & 0.27 & -0.03 & 0.02 & 0.17 & 0.3 & -0.06 & 0.73 \\
-0.21 & 0.27 & -0.18 & -0.03 & -0.54 & 0.08 & 0.47 & 0.04 & 0.58 & -0. & 0. & -0. \\
-0.01 & 0.49 & 0.23 & 0.02 & 0.59 & -0.39 & 0.29 & -0.25 & 0.23 & -0. & 0. & 0. \\
-0.04 & 0.62 & 0.22 & 0. & -0.07 & 0.11 & -0.16 & 0.68 & -0.23 & 0. & 0. & 0. \\
-0.03 & 0.45 & 0.14 & -0.01 & -0.3 & 0.28 & -0.34 & -0.68 & -0.18 & 0. & -0. & 0.
\end{pmatrix}$$

Dimension reduction, since the output is same size as original.

```
up,sp,vp=u[:,0:2],np.diag(s[0:2]),vh[:,0:2]
```

Product of (up, sp, vp): fewer 0’s.

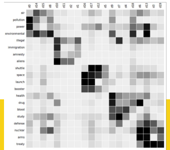
$$\Sigma = \begin{pmatrix}
3.34 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 2.54 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.35 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1.64 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.50 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1.31 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.56 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.36
\end{pmatrix}$$

words	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.32	-0.006	0.062	0.711	0.026	0.13	0.007	0.004	0.023
interface	0.241	0.007	0.018	0.632	0.008	0.098	0.038	-0.	0.002
computer	0.092	0.063	-0.143	0.76	-0.059	0.033	0.19	-0.017	-0.078
user	0.178	0.099	-0.221	1.277	-0.092	0.066	0.302	-0.027	-0.121
system	0.683	0.05	-0.026	2.057	-0.011	0.274	0.198	-0.01	-0.034
response	0.01	0.095	-0.233	0.833	-0.097	-0.002	0.275	-0.026	-0.123
time	0.01	0.095	-0.233	0.833	-0.097	-0.002	0.275	-0.026	-0.123
EPS	0.416	-0.003	0.068	0.965	0.028	0.169	0.023	0.003	0.023
survey	-0.286	0.154	-0.422	0.633	-0.175	-0.126	0.423	-0.044	-0.213
trees	-0.747	0.209	-0.624	0.005	-0.259	-0.316	0.547	-0.062	-0.305
graph	-0.935	0.269	-0.801	0.069	-0.332	-0.397	0.707	-0.08	-0.392
minors	-0.673	0.196	-0.581	0.068	-0.241	-0.286	0.515	-0.058	-0.285

$$V = \begin{pmatrix}
-0.2 & -0.61 & -0.46 & -0.54 & -0.28 & -0. & -0.01 & -0.02 & -0.08 \\
-0.06 & 0.17 & -0.13 & -0.23 & 0.11 & 0.19 & 0.44 & 0.62 & 0.53 \\
0.11 & -0.5 & 0.21 & 0.57 & -0.51 & 0.1 & 0.19 & 0.25 & 0.08 \\
-0.95 & -0.03 & 0.04 & 0.27 & 0.15 & 0.02 & 0.02 & 0.01 & -0.02 \\
0.05 & -0.21 & 0.38 & -0.21 & 0.33 & 0.39 & 0.35 & 0.15 & -0.6 \\
-0.08 & -0.26 & 0.72 & -0.37 & 0.03 & -0.3 & -0.21 & 0. & 0.36 \\
-0.18 & 0.43 & 0.24 & -0.26 & -0.67 & 0.34 & 0.15 & -0.25 & -0.04 \\
0.01 & -0.05 & -0.01 & 0.02 & 0.06 & -0.45 & 0.76 & -0.45 & 0.07 \\
0.06 & -0.24 & -0.02 & 0.08 & 0.26 & 0.62 & -0.02 & -0.52 & 0.45
\end{pmatrix}$$

$sim(human, user) = 0.89$

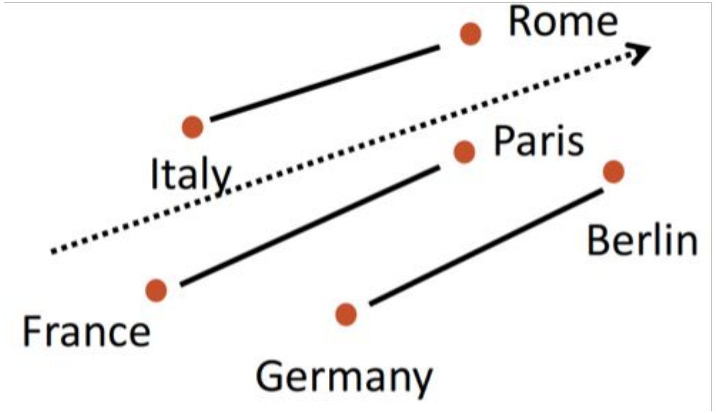
$sim(user, minors) = -0.27$



Word Embedding

Represent words using vectors:

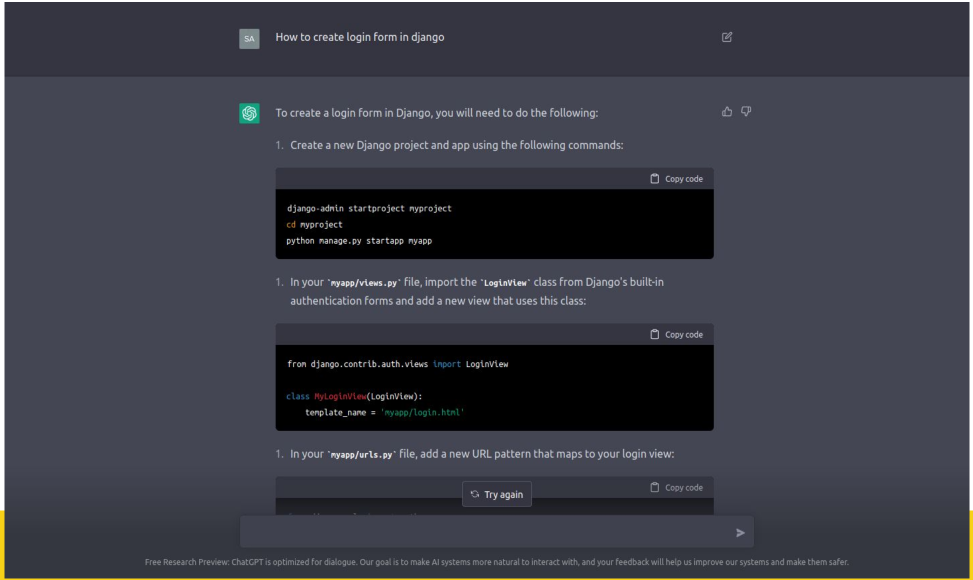
- Word → vector
- Relation between words → direction



Word2vec [Mikolov et al'13]
GloVe [Pennington et al'14]

Large Language Models (LLM)

- Machine learning models for language learnt on large-scale language datasets
- They can be used for many purposes



Supervised Learning Algorithms.

Supervised Learning Algorithms

- Linear Regression
- Decision Trees
- Support Vector Machines
- K-Nearest Neighbor
- Neural Networks

Tool	Uses	Language
Scikit-Learn	Classification, Regression, Clustering	Python
Spark MLlib	Classification, Regression, Clustering	Scala, R, Java
Weka	Classification, Regression, Clustering	Java
Caffe	Neural Networks	C++, Python
TensorFlow	Neural Networks	Python

Linear Algebra Review

Adapted from Adapted from CS229 Linear Algebra Review Fall 2022: link.

Environment Setup and Python Review

How is Python related to/with other languages?

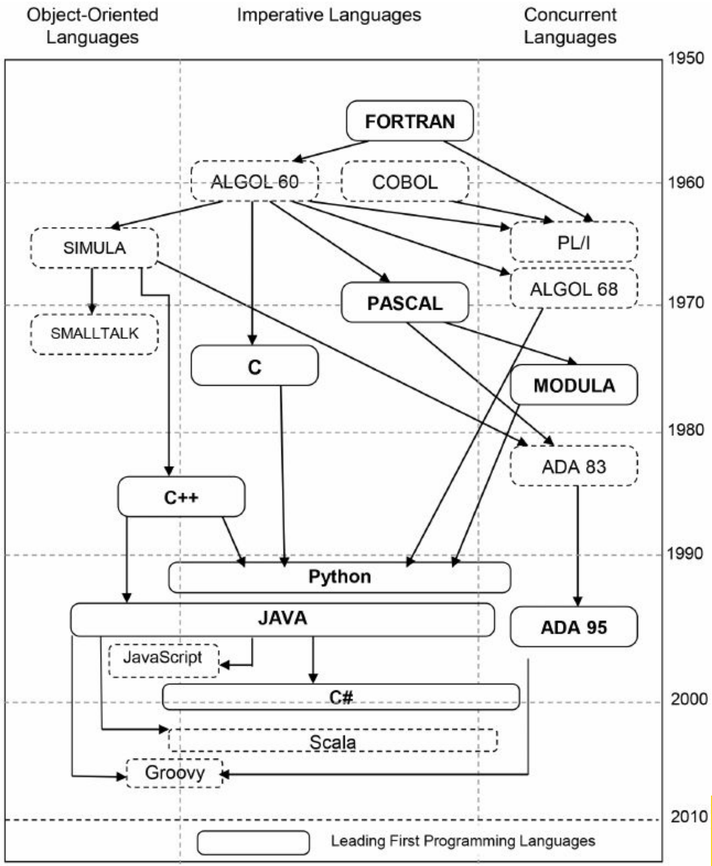
1. Python 3.6+ for this course.
2. Python can run interpreted.

Environment:

- Colab (out of box)
- Conda (mini conda) / PIP

IDE:

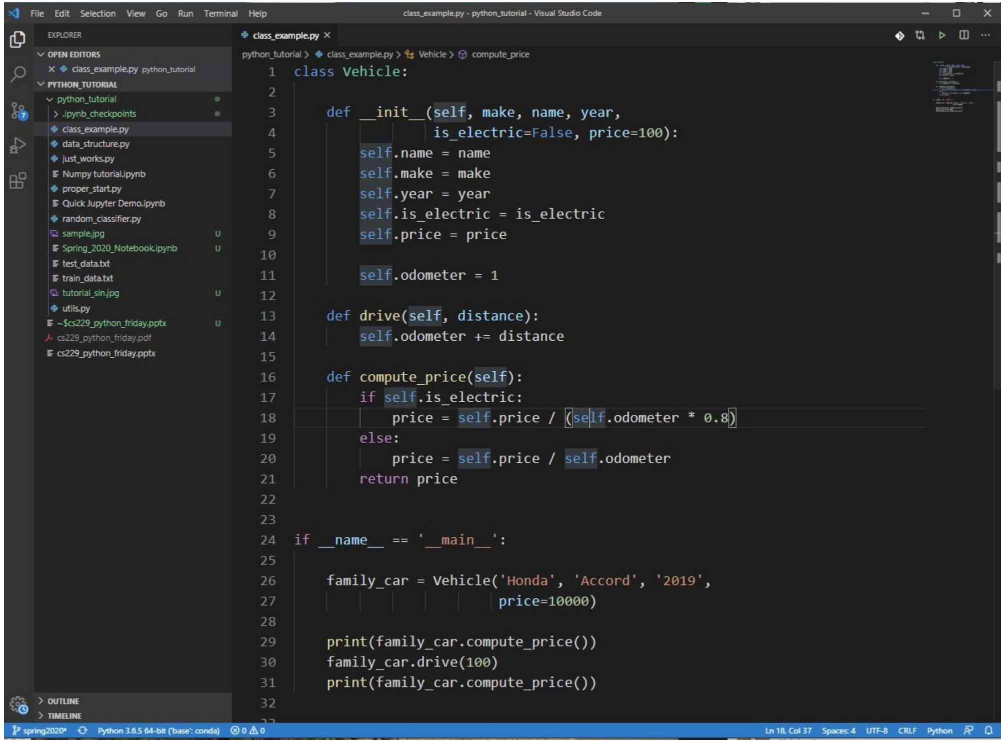
- Visual Studio Code (PyCharm, Sublime, etc.)



UCSC Silicon Valley Extension
PROFESSIONAL EDUCATION

Python IDE

- PyCharm:
  - Great debugger
  - Proper project management
  - Professional version free for students: https://www.jetbrains.com/student/
- VS Code:
  - Light weight
  - Wide variety of plugins to enable support for all languages
  - Better UI



Basic Python and Numpy

https://colab.research.google.com/drive/1Wf3iTWRDbVySM_U8byqg-w4sHi_KshSe?usp=sharing

- Python
- Numpy: package for vector and matrix manipulation. Broadcasting and vectorization saves time and amount of code
- Matplotlib: visualization library
- Pandas: dataframe (database/Excel-like) library.

Python programming:

It just works

```
def do_something(number):
    for i in number:
        print(f'Hello {i}')

do_something(5)
```

A function

Properly

```
def do_something(number):
    for i in number:
        print(f'Hello {i}')

if __name__ == '__main__':
    do_something(5)
```

What is a class/object?

Initialize the class to get an **instance** using some parameters

Instance variable

Does something with the **instance**

```python
class Vehicle:

    def __init__(self, make, name, year,
                 is_electric=False, price=100):
        self.name = name
        self.make = make
        self.year = year
        self.is_electric = is_electric
        self.price = price

        self.odometer = 0

    def drive(self, distance):
        self.odometer += distance

    def compute_price(self):
        if self.is_electric:
            price = self.price / (self.odometer * 0.8)
        else:
            price = self.price / self.odometer
        return price
```

How use a class/object

Instantiate a class,

get an **instance**

Call an instance method

```
if __name__ == '__main__':

    family_car = Vehicle('Honda', 'Accord', '2019',
                         price=10000)

    print(family_car.compute_price())
    family_car.drive(100)
    print(family_car.compute_price())
```