CSC380: Principles of Data Science

Final Project

Kyoungseok Jang

Announcements

- HW7, final project out
- HW5 solution uploaded

Overview

- Participate in NBME task
  - National Board of Medical Examiner
- Mostly guided problem solving, but some open-ended questions
- Extra points for those who achieve high score in the leaderboard.

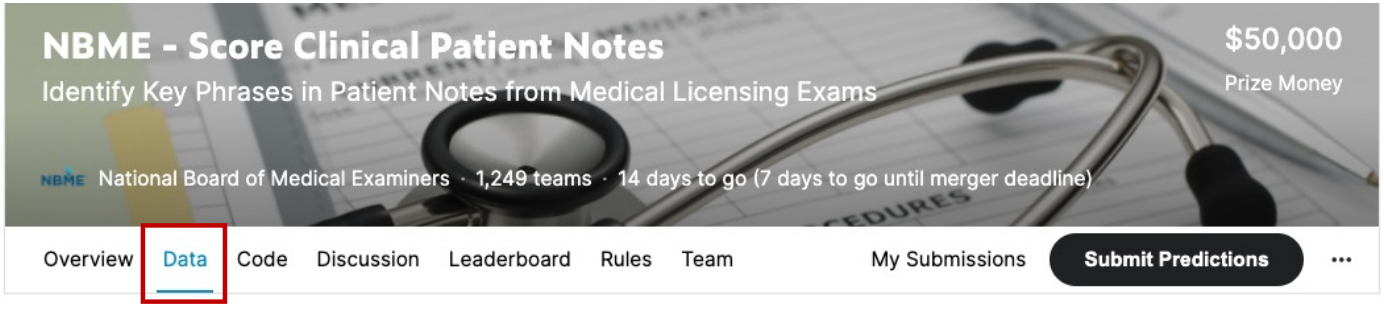
Problem 1. Familiarize yourself with NBME

- NBME task
  - Clinical skill examination
  - Exam takers interact with patients to write patient note (without missing important information; “features”)
  - Want: **Automatically grade patient notes**! (or, semi-automatically)

- Total 10 types of patients (called “case”)

Problem 1. Familiarize yourself with NBME

Understand data



click

The data:

patient_notes.csv
features.csv

For training:

train.csv

For submitting your ML code for evaluation:

test.csv
sample_submission.csv

Problem 1. Familiarize yourself with NBME

patient_notes.csv

id for patient_note 0 to 9

# pn_num	# case_num	A pn_history
A unique identifier for each patient note.	A unique identifier for the clinical case a patient note represents.	The text of the encounter as recorded by the test taker.
 0 95.3k	 0 9	42146 unique values
00000	0	17-year-old male, has come to the student health clinic complaining of heart pounding. Mr. Cleveland...
00001	0	17 yo male with recurrent palpitations for the past 3 mo lasting about 3 - 4 min, it

Exam taker's answers for each case

patient_notes.csv

Example patient notes

"17-year-old male, has come to the student health clinic complaining of heart pounding. Mr. Cleveland's mother has given verbal consent for a history, physical examination, and treatment

-began 2-3 months ago,sudden,intermittent for 2 days(lasting 3-4 min),worsening,non-allev/aggrav

-associated with dispnea on exersion and rest,stressed out about school

-reports fe feels like his heart is jumping out of his chest

-ros:denies chest pain,dyaphoresis,wt loss,chills,fever,nausea,vomiting,pedal edeam

-pmh:non,meds :aderol (from a friend),nkda

-fh:father had MI recently,mother has thyroid dz

-sh:non-smoker,mariguana 5-6 months ago,3 beers on the weekend, basketball at school

-sh:no std"

17yo male with no pmh here for evaluation of palpitations. States for the last 3-4mo he has felt that his heart with intermittently "beat out of his chest," with some associated difficulty catching his breath. States that the most recent event was 2 days ago, and during activity at a soccer game. He does not seem to note any specific precipitatinig factors at this time. He also states that he feels as if he will faint during these events, but has not lost consciousness at any point. Furthermore, he does endorse theses attacks occuring 1-2 times a month and peak at 4 mins. He denies any stressors at home. ROS: denies weight loss, fevers, recnet illness, change in bowel habits. PMH: negative, PSH negative, FHX mom with thyroid disorder, dad with heart condition and MI at 52yo. SHX no tobacco, ETOH on weekends, Marijuana tried a month ago. Med: is taking some of roommates Adderoll intermittently (last was 2 days ago prior to event). KNDA

patient_notes.csv

"Dillon Cleveland is a 17 year old male with no signfiicant past medical history presenting today with ""heart pounding"" for the past 2-3 months. He first noticed an episode when he was sitting down and has had 5-6 over the past 3 months. In the most recent episode has felt light headed and had to sit down while playing basketball. There does not appear to be any precipating factors for there episodes, and has never lost consciousness with them. He has not had any changes in his bowel habits or sleep. No sensitivity to heat or cold. No Weight gain or loss.

Medical history: None
Surgical history: None
Medications: Adderal (non-prescription)
Allergies: NKA
Family history: Father had MI 1 year ago. No history of arrythmias. Mother has a thyroid issue.
SocNo alcohol or tobacco. Tried marijuana once. Drinks one cup of coffee daily. Takes 2x adderal per week for the previous 8 months."

A variety of writing style!!

You have a bunch of these for each case_num (=patient).

There are total 10 actual patients.

Note typos

features.csv

Features are not the ‘feature vector’ we learned! From now on, I will call this ‘evaluation’ features.

# feature_num A unique identifier for each feature. 0 916	# case_num The case to which this patient note belongs. 0 9	A feature_text A description of the feature. Female 5% Male 2% Other (133) 93%
000	0	Family-history-of-MI-OR-Family-history-of-myocardial-infarction
001	0	Family-history-of-thyroid-disorder
002	0	Chest-pressure
...		
010	0	Few-months-duration
011	0	17-year
012	0	Male
101	1	Weight-loss
102	1	Not-sexually-active

Features: the evaluation target. The first number for the feature_num seems to be case_num

Each case_num has a different number of features.

Visit the Kaggle NBME page to see more on the data

Goal of the competition

The goal of this competition: have a software that, given a patient_note, automatically highlights the relevant features.

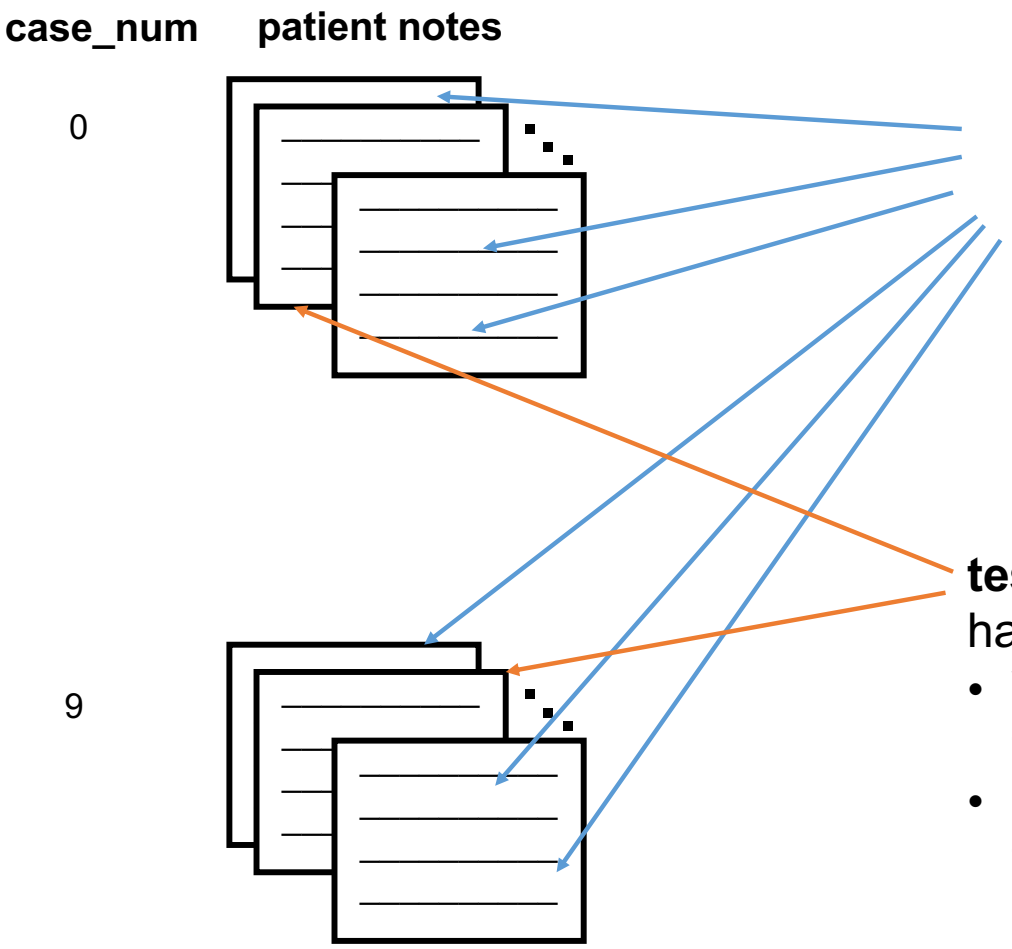
Feature 9: Heart pounding

"Dillon Cleveland is a 17 year old male with no signfiicant past medical history presenting today with ""heart pounding"" for the past 2-3 months. He first noticed an episode when he was sitting down and has had 5-6 over the past 3 months. In the most recent episode has felt light headed and had to sit down while playing basketball. There does not appear to be any precipating factors for there episodes, and has never lost consciousness with them. He has not had any changes in his bowel habits or sleep. No sensitivity to heat or cold. No Weight gain or loss.

Medical history: None
Surgical history: None
Medications: Adderal (non-prescription)
Allergies: NKA
Family history: Father had MI 1 year ago. No history of arrythmias. Mother has a thyroid issue.
SocNo alcohol or tobacco. Tried marijuana once. Drinks one cup of coffee daily. Takes 2x adderal per week for the previous 8 months."

Feature 0: family history of MI

Data Overview



Annotations in **train.csv**:
Highlights relavant portion of the note for each evaluation features.

test.csv points out which ones you will have to make prediction

- You will have to highlight portions of texts for each evaluation feature.
- Of course, these never appear in train.csv!

train.csv

- train.csv contains those annotations for a subset of patient notes.
- Many columns, but the key is

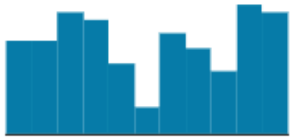
(case_num, patient note number, evaluation feature num, locations)

E.g., (0, 16, 4, ['321 329', '404 413', '652 661’])

- Other columns are not essential; just for your convenience.

train.csv

So, train.csv contains those annotations for a subset of patient notes.

# case_num The case to which this patient note belongs.	# pn_num The patient note annotated in this row.	# feature_num The feature annotated in this row.	A annotation The text(s) within a patient note indicating a feature. A feature may be indicated multiple times within a single note.	A location Character spans indicating the location of each annotation within the note.
 0 9	 16 95.3k	 0 916	[] 31% ['F'] 2% Other (9597) 67%	[] 31% ['0 5'] 1% Other (9719) 68%
0	00016	000	['dad with recent heart attcak']	['696 724']
0	00016	001	['mom with "thyroid disease']	['668 693']
0	00016	002	['chest pressure']	['203 217']
0	00016	003	['intermittent episodes', 'episode']	['70 91', '176 183']

python indexing!
('start end' means from start to end-1)

train.csv

Be aware:

# pn_num The patient note annotated in this row.	# feature_num The feature annotated in this row.	A annotation The text(s) within a patient note indicating a feature. A feature may be indicated multiple times within a single note.	A location Character spans indicating the location of each annotation within the note.
00082	009	['heart pounding', 'heart racing', 'heart pounding']	['85 99', '126 138', '126 131;143 151']

some annotations are not contiguous.

E.g.) Heart (……) pounding

some even overlap with other annotation.

train.csv

So, you need to use ML to learn from train.csv and be able to mark ‘location’.

[Prediction task]

- Given: (case_num, pn_num)
- For each evaluation feature f
  - For each feature you need to perform prediction of those ‘locations’ for each feature

{0,…,12} for case_num=0

[Training]

- For each case_num c ∈ $\{0, \ldots, 9\}$
  - Use the annotations in train.csv to train a function $g_c(\text{pn\_history})$ that returns a list of locations, each corresponding to an evaluation feature.

test.csv

This will be replaced to a large number of other entries when you submit your answer to Kaggle!

id	case_num	pn_num	feature_num
5 unique values	5 total values	5 total values	5 total values
00016_000	0	00016	000
00016_001	0	00016	001
00016_002	0	00016	002
00016_003	0	00016	003
00016_004	0	00016	004

sample_submission.csv

Your code will have to write answers like this:

id Unique identifier for this instance, a feature within a patient note.	location Character spans indicating the location(s) of the feature within the note.
5 unique values	[null] 40% 0 100 20% Other (2) 40%
00016_000	0 100
00016_001	
00016_002	200 250;300 400
00016_003	
00016_004	75 110

each row corresponds to each row in test.csv

Evaluation criteria

Example

Suppose we have an instance:

```
| ground-truth | prediction    |
|--------------|---------------|
| 0 3; 3 5     | 2 5; 7 9; 2 3 |
```

These spans give the sets of indices:

```
| ground-truth | prediction |
|--------------|------------|
| 0 1 2 3 4    | 2 3 4 7 8  |
```

We therefore compute:

- `TP = size of {2, 3, 4} = 3`
- `FN = size of {0, 1} = 2`
- `FP = size of {7, 8} = 2`

Repeat for all instances, collect the TPs, FNs, and FPs, and compute the final F1 score.

First, sum up all the TP, FN, and FP computed from every row of submission.csv

Call these summed values as TPs, FNs, and FPs.

precision = TPs / (TPs + FPs)
recall = TPs / (TPs + FNs)

F1 = 2*precision*recall / (precision + recall)
= 2*TPs / (2*TPs + FPs + FNs)

This is called micro-F1 score!

How to use ML

Example: “One time durign a baskeball game two days ago light headedness, pressure in the chest, but no fainting”

Split into words (record their locations as well):

[‘one’, ‘time’, …, ‘pressure’, ‘in’, ‘the, ‘chest’, ’catching’, …]

Idea: what if we build a classifier at the word level?

For each word location [start, end]

- feature vector: [word length, freq(‘a’), freq(‘b’), …, freq(‘z’), freq(‘0’), …, freq(‘9’)]
  e.g., for ‘pressure’: freq(‘a’) is 0, freq(‘p’) is 1/8, freq(‘s’) is 1/4.
- label: whether the location [start, end] belongs to the annotation.

Note: ‘pressure’ may appear twice with different label! (‘heart’ pressure yes, ‘air’ pressure no)

How to use ML

For making prediction:

- Split by words as before, make prediction for each (+/-)
- Collect the character locations predicted as positive.

Example prediction:

"Most recent episode was accompanied by chest pressure and lasted 10 minutes."

ground truth is "chest pressure" so you get it right except for one space!

Another one:

"Most recent episode was accompanied by pressure in the chest and lasted 10 minutes."

- Q: 'in' and 'the' received positive label. Why wouldn't it be classified as positive?

How to use ML: Two Words

Observation: Often, words are meaningful ‘phrase-wise’.
Idea: Let’s consider **two words** at a time!

Example: “One time durign a baskeball game two days ago light headedness, pressure in the chest, but no fainting”

Extract (and record their locations):

[‘one time’, ‘time durign’, …, ‘headedness, pressure’, ‘pressure in’, ‘in the’, ‘the chest’, chest, but’, …]

For each two words location [start, end]

- feature vector: [n_characters, freq(‘a’), freq(‘b’), …, freq(‘z’), freq(‘0’), …, freq(‘9’)]
- label: whether the location [start, end] entirely belongs to the annotation.

How to use ML: Two Words

For making prediction:

- Split by two words as before, make classification
- Collect the character locations predicted as positive

// locations may overlap => just take 'logical or'

Example prediction:

"Most recent episode was accompanied by pressure in the chest and lasted 10 minutes."

- 'pressure in' => +
- 'in the' => -
- 'the chest' => +

But some expressions may be meaningful by just one word… what would you do?

How to use ML: Combined

Idea: let's use both 'one word' and 'two words' classifiers!

- Option 1: Build two separate classifiers
- Option 2: Build one classifier

  How come?

  Recall that the feature vector is [n_characters, freq('a'), freq('b'), …, freq('z'), freq('0'), …, freq('9')]..!

  $\Rightarrow$ Pool all one-word-based data points and two-word-based ones, and train!

  In fact, I would add the number of words as a feature as well:

  [**n_words**, n_characters, freq('a'), freq('b'), …, freq('z'), freq('0'), …, freq('9')]

I like option 2 better: if n_words mattered, the classifier will pick up that information. Otherwise, it wouldn't.

- E.g., if 'to me' and 'tome' have a different labels, decision tree would use n_words to distinguish it! Otherwise, the decision tree may not even use n_words.

How to use ML: Combined

But how to make predictions?

Given a text

- extract all {(location, word)}, and {(location, two words)}
- make predictions
- collect all positive locations, take ‘logical or’ (i.e., union of all positive locations)

E.g., if [100,110] and [105,120], then output [100,120]

How to use ML: Combined

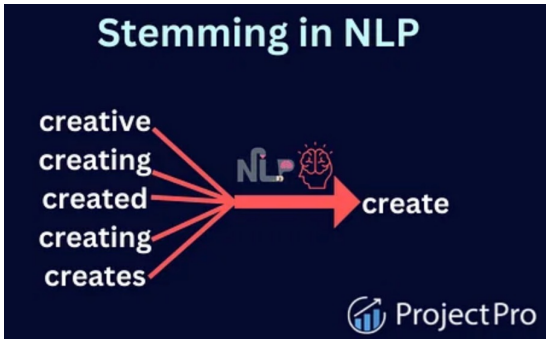
But why stop at two words?

In my code provided to you, I use up to 5 words.

- There is a parameter 'W' in the code set to 5.
- Feel free to change it around.

Note that there are many other choices to make!

- E.g., which characters to take into account // rules to separate words (e.g., 'a/b' one word or two words?) // 'stemming' // …



Useful python functions

eval(*expression*[, *globals*[, *locals*]])

The arguments are a string and optional globals and locals. If provided, *globals* must be a dictionary. If provided, *locals* can be any mapping object.

The *expression* argument is parsed and evaluated as a Python expression (technically speaking, a condition list) using the *globals* and *locals* dictionaries as global and local namespace. If the *globals* dictionary is present and does not contain a value for the key `__builtins__`, a reference to the dictionary of the built-in module `builtins` is inserted under that key before *expression* is parsed. That way you can control what builtins are available to the executed code by inserting your own `__builtins__` dictionary into *globals* before passing it to `eval()`. If the *locals* dictionary is omitted it defaults to the *globals* dictionary. If both dictionaries are omitted, the expression is executed with the *globals* and *locals* in the environment where `eval()` is called. Note, *eval()* does not have access to the nested scopes (non-locals) in the enclosing environment.

The return value is the result of the evaluated expression. Syntax errors are reported as exceptions. Example:

```
>>> x = 1
>>> eval('x+1')
2
```

Note that we use micro F1 score.

In this case, averaging precision/recall may not be a reasonable thing to do. The # of TPs,FPs,FNs could be quite different across the folds.

E.g., precision 1: 1/(1+5) = 1/6, precision 2: 10/(10+2) = 5/6

average: 0.5

micro precision: 11/(11 + 7) = 0.61

Cross Validation

In this case, here is a preferred way to compute precision using cross validation.

Gather TP/FP/FN for each fold, using cross validation.

Add up TP/FP/FN and then compute precision

(do the same for recall)

This is what the provided code does.

CSC380: Principles of Data Science

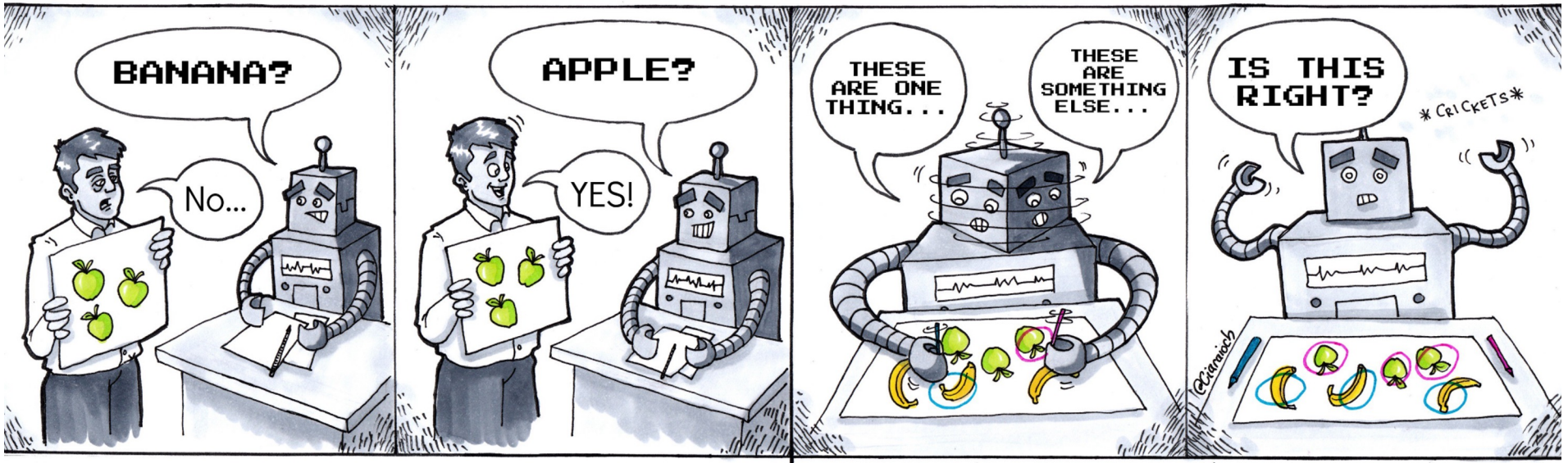
k-means Clustering

Kyoungseok Jang

Slides are in part from Enfa George (TA in 2021)

What is unsupervised learning?

- Learning with unlabeled data
- What can we expect to learn?
  - **Clustering**: obtain partition of the data that are well-separated.
    - can be viewed as a preliminary classification without predefined class labels.
  - **Components**: extract common components that compose data points.
    - e.g., topic modeling given a set of articles: each article talks about a few topics => extract the topics that appear frequently.
- Use
  - As a summary of the data
    - **Exploratory data analysis**: what are the **patterns** we can get even without labels?
  - Often used as a 'preprocessing techniques'
    - e.g., extract useful **features** using soft clustering assignments (e.g., "gaussian mixture model")



Supervised Learning

Unsupervised Learning

Task 1 : Group These Set of Document into 3 Groups based on meaning

Doc1 : Health , Medicine, Doctor

Doc 2 : Machine Learning, Computer

Doc 3 : Environment, Planet

Doc 4 : Pollution, Climate Crisis

Doc 5 : Covid, Health , Doctor

Task 1 : Group These Set of Document into 3 Groups based on meaning

Doc1 : Health , Medicine, Doctor

Doc 2 : Machine Learning, Computer

Doc 3 : Environment, Planet

Doc 4 : Pollution, Climate Crisis

Doc 5 : Covid, Health , Doctor

Task 1 : Group These Set of Document into 3 Groups based on meaning

Doc1 : Health , Medicine, Doctor

Doc 5 : Covid, Health , Doctor

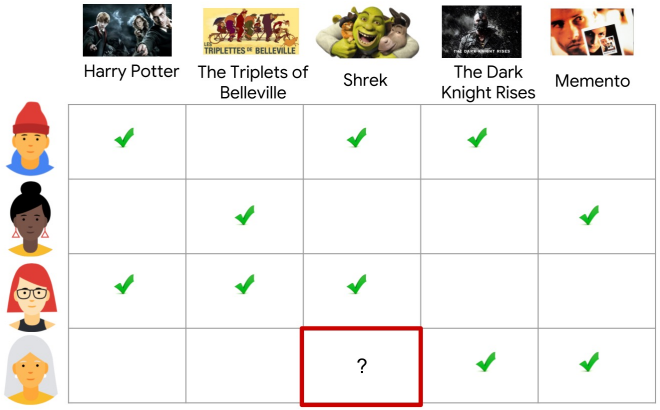
Doc 3 : Environment, Planet

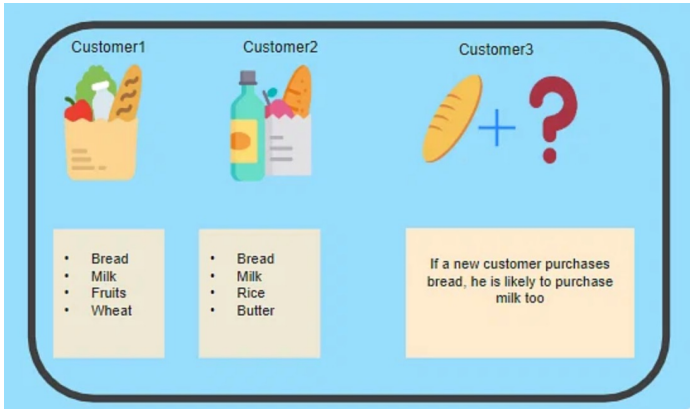
Doc 4 : Pollution, Climate Crisis

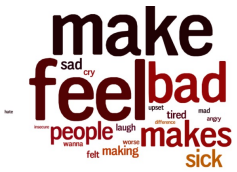
Doc 2 : Machine Learning, Computer

Task 2: Recommendation

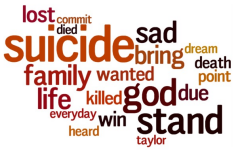
- Discover the probability of the co-occurrence of items in a collection
  - Market basket analysis
  - Semantic clustering (Topic modeling)
  - Movie recommendation







"feelings"

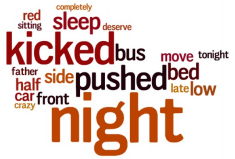


"suicide"

"family"

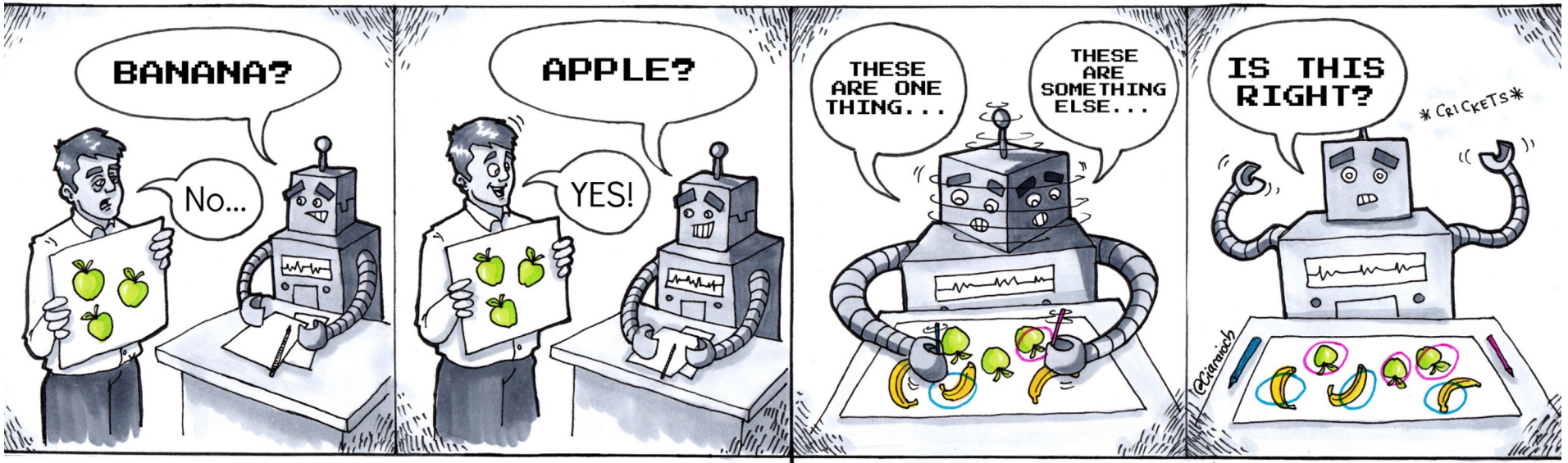
"school"

"verbal bullying"



"physical bullying"

From: https://www.simplilearn.com/tutorials/machine-learning-tutorial/supervised-and-unsupervised-learning
And https://developers.google.com/machine-learning/recommendation/collaborative/basics



Supervised Learning

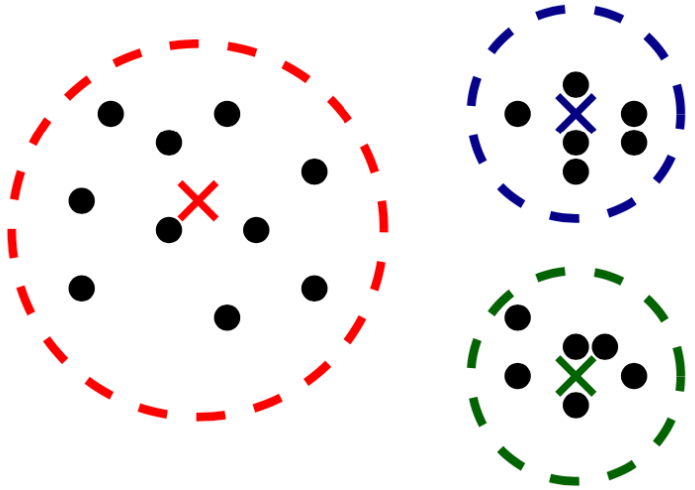
Unsupervised Learning

Clustering

- Input: k: the number of clusters (hyperparameter)

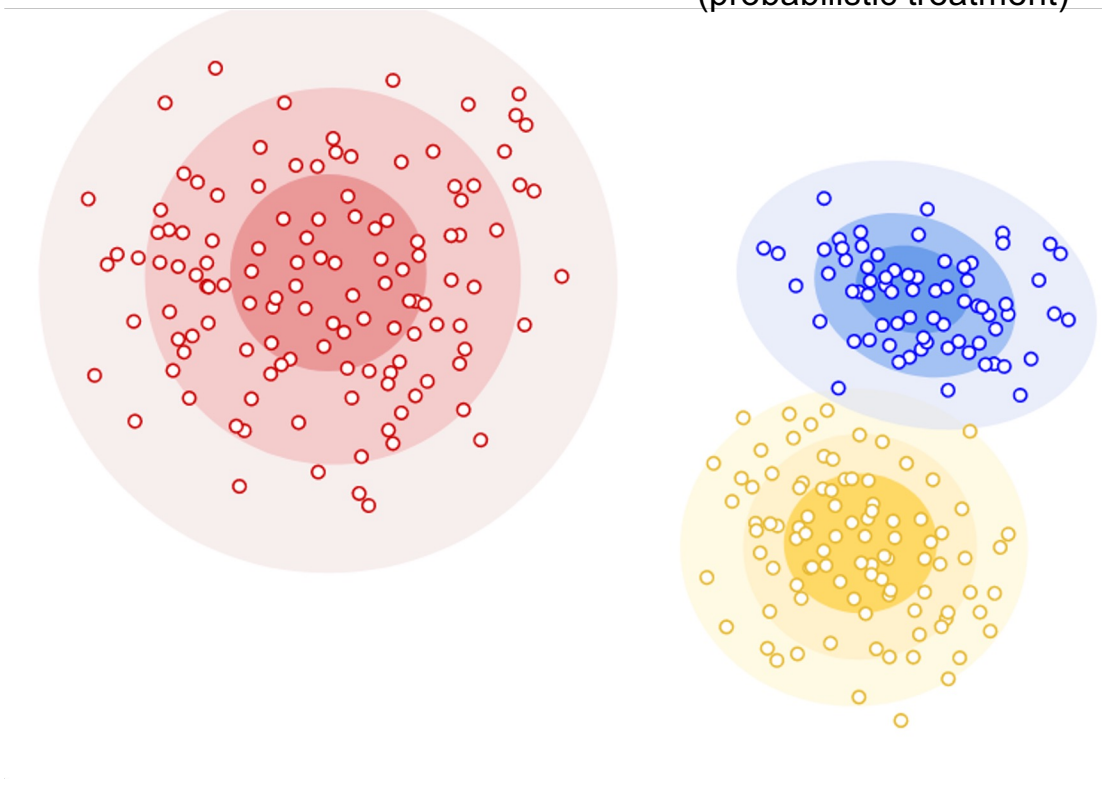
  $S = \{x_1, \ldots, x_n\}$

- Output
  - partition $\{G_i\}_{i=1}^{k}$ s.t. $S = \cup_i G_i$ (disjoint union).
  - often, we also obtain 'centroids'

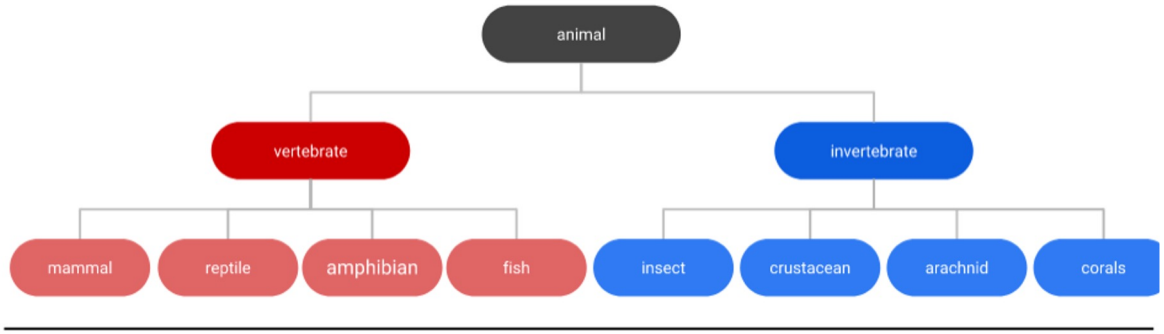


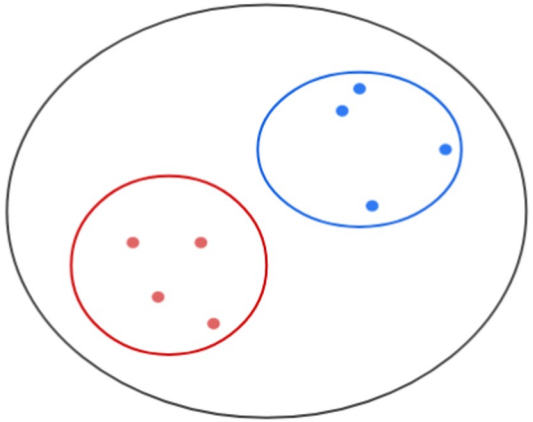
Distribution-based Clustering

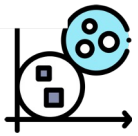
(probabilistic treatment)



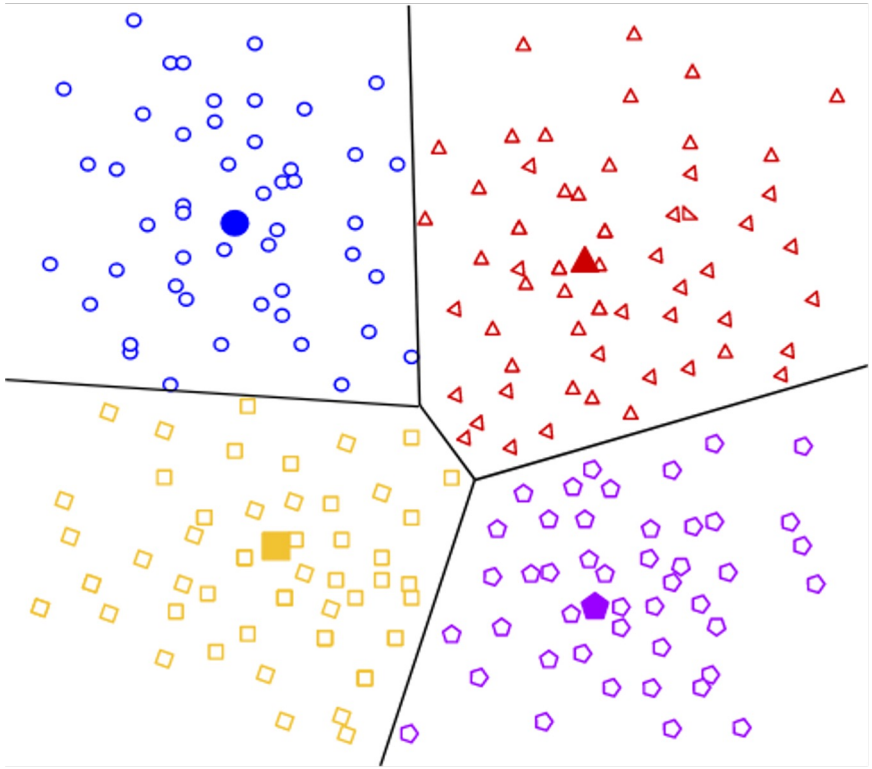
Hierarchical Clustering







Centroid-based Clustering

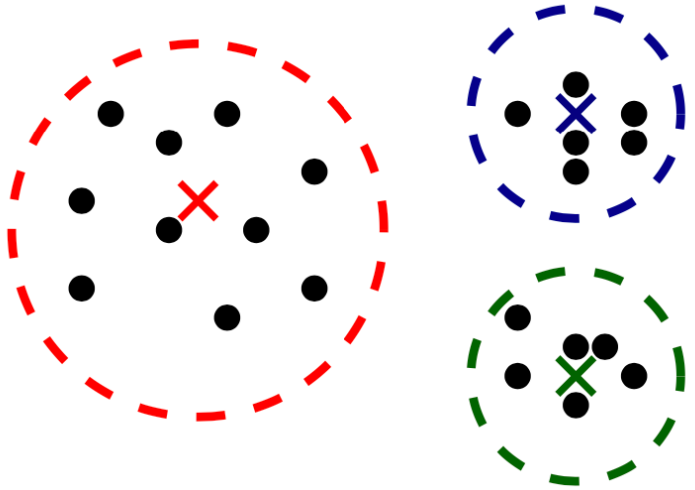


Clustering

- Input: k: the number of clusters (hyperparameter)

$$S = \{x_1, \dots, x_n\}$$

- Output
  - partition $\{G_i\}_{i=1}^{k}$ s.t. $S = \cup_i G_i$ (disjoint union).
  - often, we also obtain 'centroids'



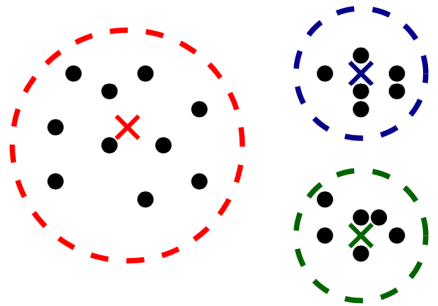
- Q: if we are given the groups, what would be a reasonable definition of centroids?
  - The **point** that has the minimum average **distance** to the datapoints?
  - The **datapoint** that has the minimum average **distance** to the datapoints?
  - The **point** that has the minimum average **squared distance** to the datapoints?

=> Turns out, the last one corresponds to the average point!

k-means clustering

- Idea: if someone gave us k reasonable centroids $c_1, \ldots, c_k$, we can partition the data with them.

$$A(x) = \arg\min_{j \in [k]} \|x - c_j\|_2^2$$



- But we don't have those centroids ⇒ Let's find them with an optimization formulation.

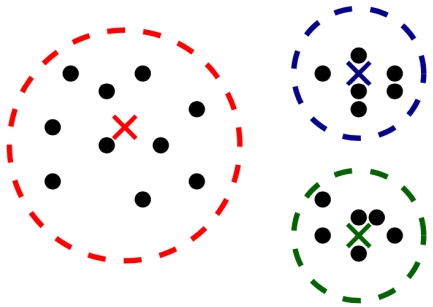
$$\arg\min_{c_1,\ldots,c_k} \sum_{i=1}^{n} \|x_i - A(x_i)\|_2^2 = \arg\min_{c_1,\ldots,c_k} \sum_{i=1}^{n} \min_{j \in [k]} \|x_i - c_j\|_2^2$$

=> NP-hard

k-means Clustering

Lloyd's algorithm: solve it approximately (heuristic)

(but people just say it is k-means clustering algorithm)



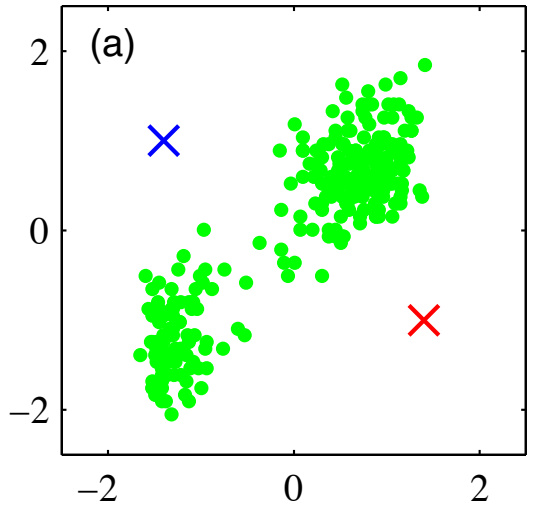
(1957 by Stuart P. Lloyd but independently developed by Joel Max in 1960)

Observation: The chicken-and-egg problem.

- If you knew the cluster assignments… just find the centroids as the average
- If you knew the centroids… make cluster assignments by the closest centroid.

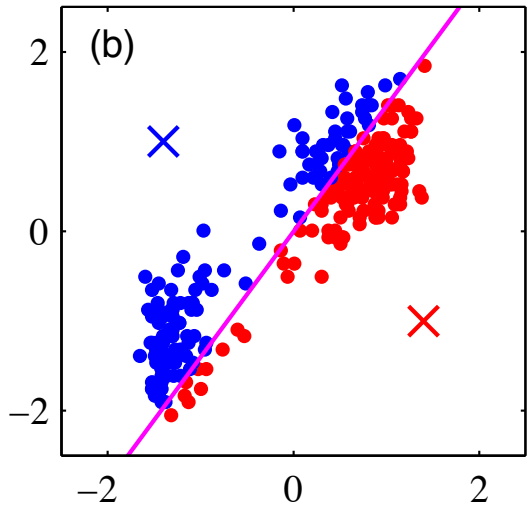
Why not: start from some centroids and then alternate between the two?

Initialization

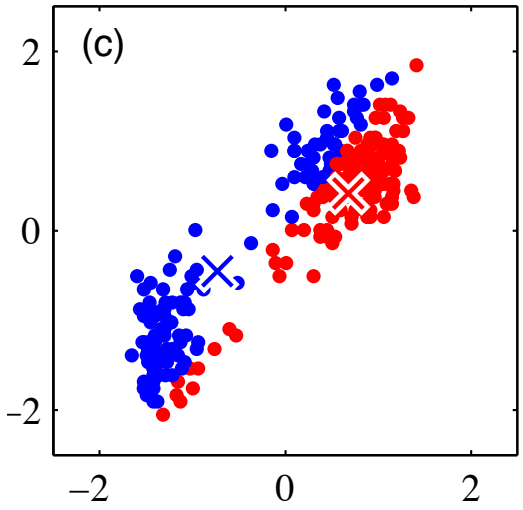


Arbitrary/random initialization of c_1 and c_2

Iteration 1

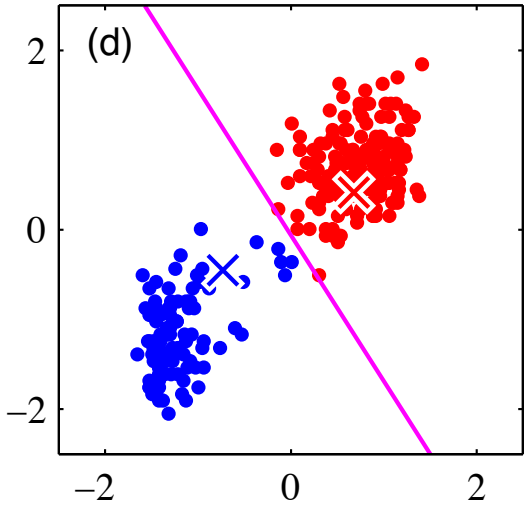


(A) update the cluster assignments.

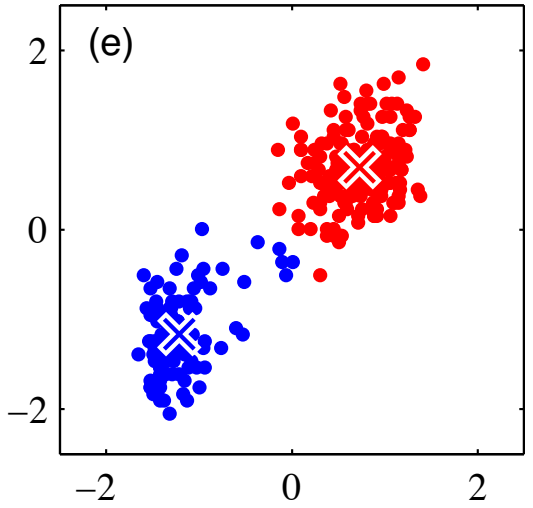


(B) Update the centroids $\{c_j\}$

Iteration 2

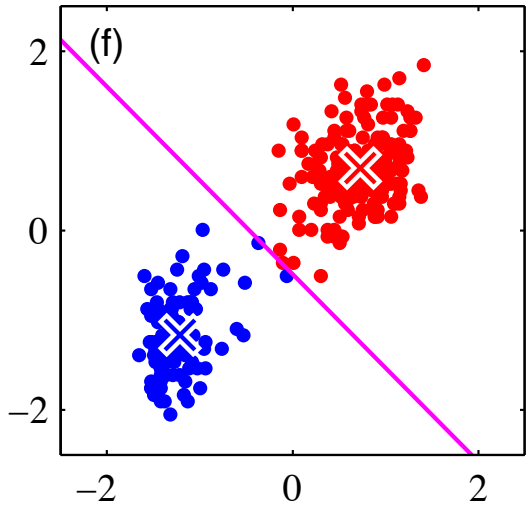


(A) update the cluster assignments.

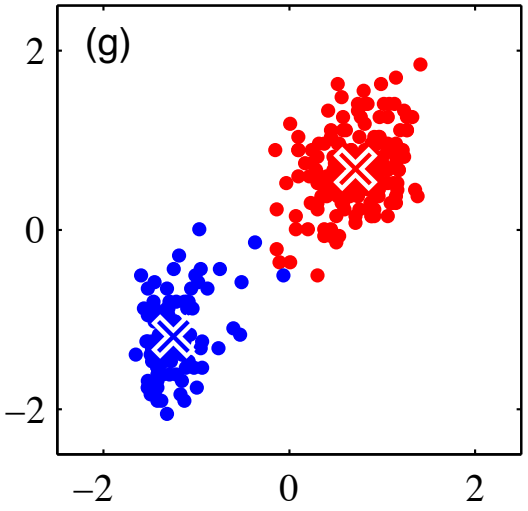


(B) Update the centroids $\{c_j\}$

Iteration 3

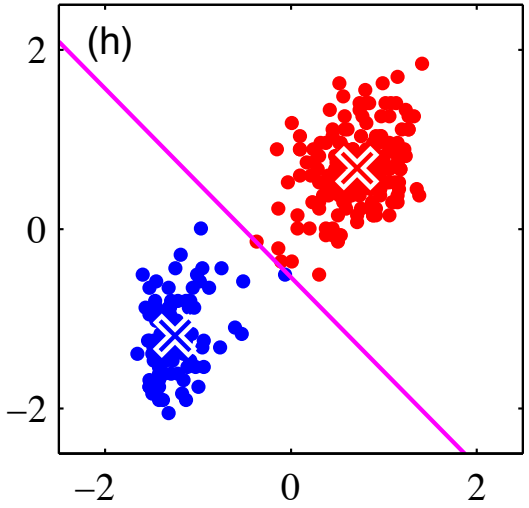


(A) update the cluster assignments.

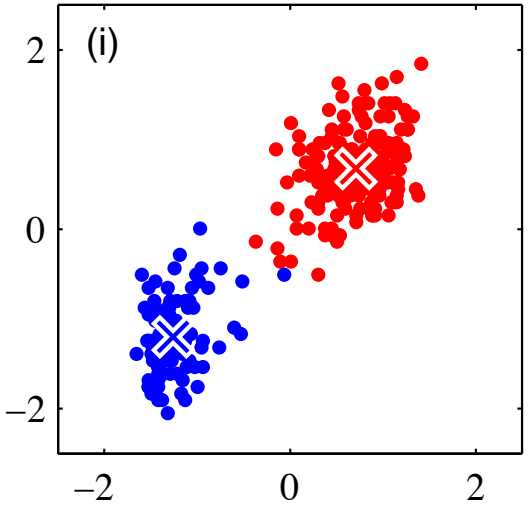


(B) Update the centroids $\{c_j\}$

Iteration 4

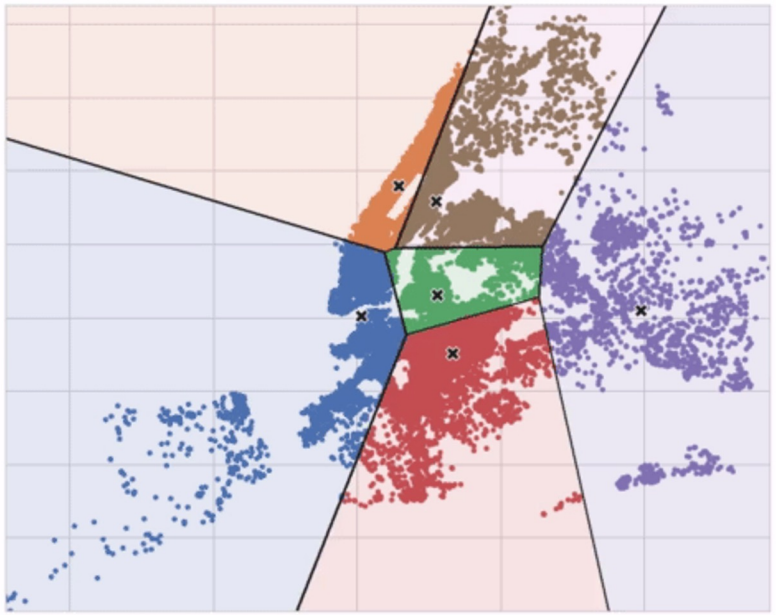


(A) update the cluster assignments.



(B) Update the centroids $\{c_j\}$

Iterating until Convergence



Animation from Kaggle

k-means clustering

Input: k: num. of clusters, $S = \{x_1, \dots, x_n\}$

[Initialize] Pick $c_1, \dots, c_k$ as randomly selected points from S (see next slides for alternatives)

For t=1,2,…,max_iter

- ***[Assignments]*** $\forall x \in S, \quad a_t(x) = \arg\min_{j \in [k]} \|x - c_j\|_2^2$
- If $t \neq 1$ AND $a_t(x) = a_{t-1}(x), \forall x \in S$
  - break
- ***[Centroids]*** $\forall j \in [k], \quad c_j \leftarrow \text{average}(\{x \in S : a_t(x) = j\})$

Output: $c_1, \dots, c_k$ and $\{a_t(x_i)\}_{i \in [n]}$

But,

It may converge to a local rather than global minimum.

$$\arg \min_{c_1,\ldots,c_k} \sum_{i=1}^{n} \min_{j\in[k]} \left\| x_i - c_j \right\|_2^2$$

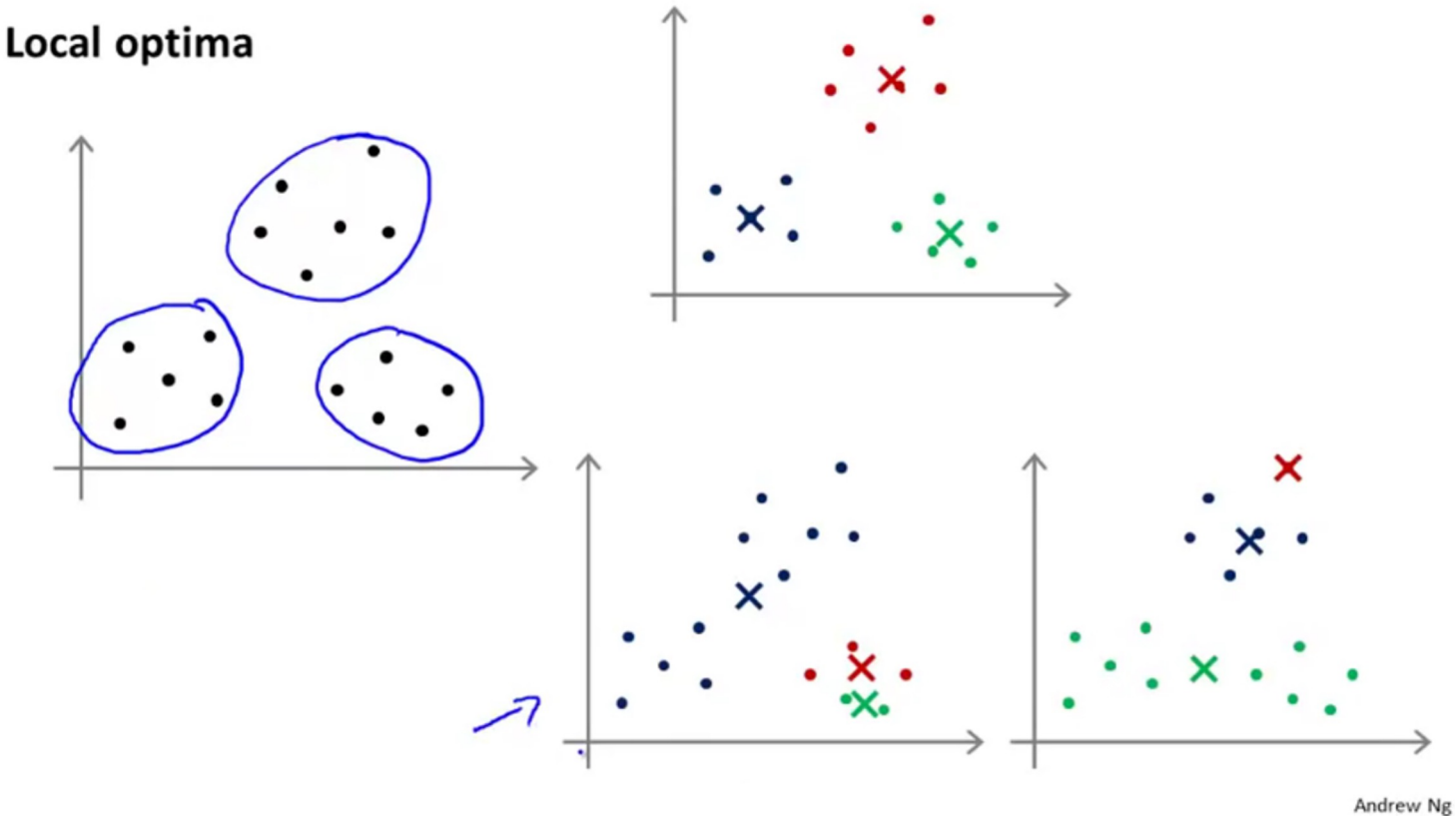


Image from Andrew NG Coursera Machine Learning Course

Issue 1: Unreliable solution

- You usually get suboptimal solutions
- You usually get different solutions every time you run.

- **Standard practice**: Run it 50 times and take the one that achieves the smallest objective function
  - Recall: $\min_{c_1,\dots,c_k} \sum_{i=1}^{n} \min_{j\in[k]} \|x_i - c_j\|_2^2$

Each run of algorithm outputs $c_1, \dots, c_k$.
Compute this to evaluate the quality!

- And/or, change the initialization (next slide)
  - Idea: ensure that we pick a widespread $c_1, \dots, c_k$

Two alternative initializations.

- **Farthest-first traversal** ⇒ Sequentially choose c_j that are the farthest from the previously-chosen.
  - Pick $c_1 \in \{x_1, \dots, x_n\}$ arbitrarily (or randomly)
  - For $j = 2, \dots, k$
    - Pick $c_j \in \mathbb{R}^d$ as a point in $\{x_1, \dots, x_n\}$ that maximizes the squared distances to $c_1, \dots, c_{j-1}$.

$$c_j = \arg\max_{i \in [n]} \min_{j'=1,\dots,j-1} \|x_i - c_{j'}\|_2^2$$

- ***k*-means++**
  - Pick $c_1 \in \{x_1, \dots, x_n\}$ uniformly at random
  - For $j = 2, \dots, k$
    - Define a distribution $\forall i \in [n]$, $\mathbb{P}(c_j = x_i) \propto \min_{j'=1,\dots,j-1} \|x_i - c_{j'}\|_2^2$
    - Draw c_j from the distribution above.

More likely to choose x_i that is farthest from already-chosen centroids.

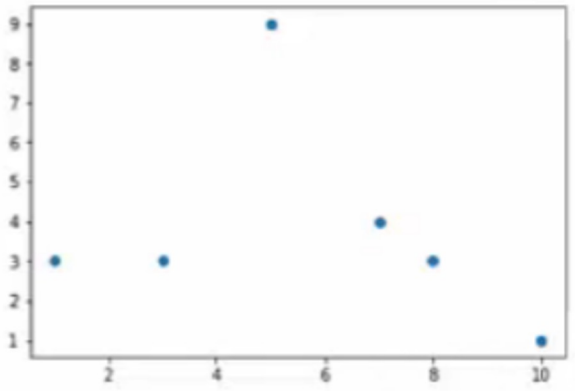
=> has a mathematical guarantee that it will be better than an arbitrary starting point!

Suppose we have the small dataset

- [(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.

We begin by randomly selecting (7,4) to be a cluster center.

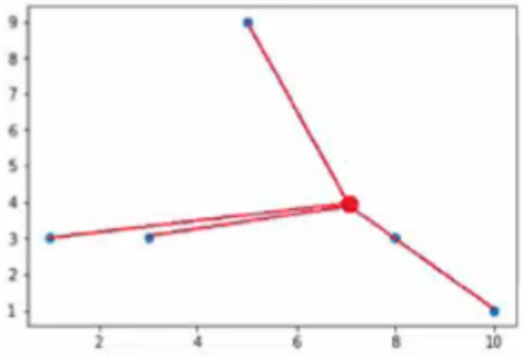
x	$\min(d(x, z_i)^2)$
(7,4)	
(8,3)	
(5,9)	
(3,3)	
(1,3)	
(10,1)	



From Sara Jensen's Youtube Channel

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We begin by randomly selecting (7,4) to be a cluster center.

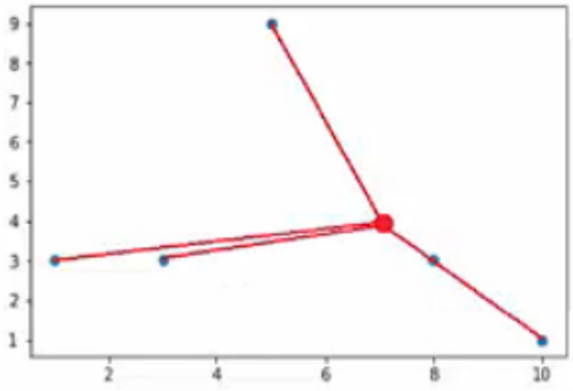
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	2
(5,9)	29
(3,3)	17
(1,3)	37
(10,1)	18



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We begin by randomly selecting (7,4) to be a cluster center.

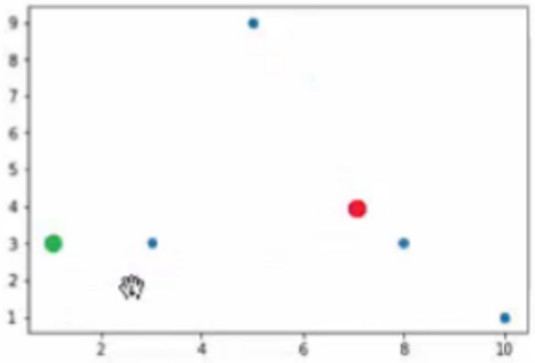
x	prob
(7,4)	-
(8,3)	2/103
(5,9)	29/103
(3,3)	17/103
(1,3)	37/103
(10,1)	18/103



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We add (1,3) to the list of cluster centers.

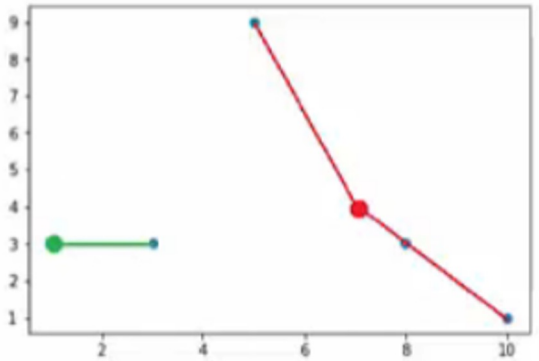
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	
(5,9)	
(3,3)	
(1,3)	-
(10,1)	



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We add (1,3) to the list of cluster centers.

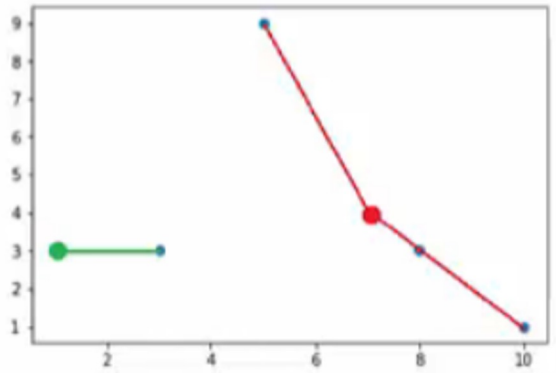
x	$\min(d(x, z_i)^2)$
(7,4)	-
(8,3)	2
(5,9)	29
(3,3)	4
(1,3)	-
(10,1)	18



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We add (1,3) to the list of cluster centers.

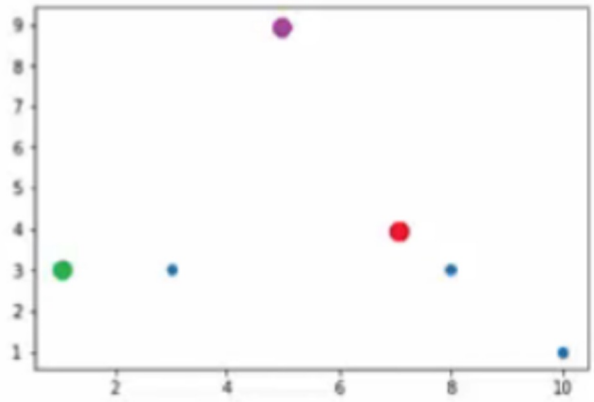
x	prob
(7,4)	-
(8,3)	2/55
(5,9)	29/55
(3,3)	4/55
(1,3)	-
(10,1)	18/55



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We add (5,9) to the list of cluster centers.

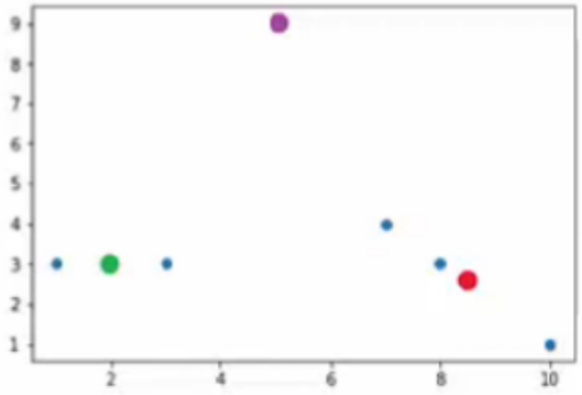
x	prob
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)	



[From Sara Jensen's Youtube Channel]

Suppose we have the small dataset
[(7,4),(8,3),(5,9),(3,3),(1,3),(10,1)] to which we wish to assign 3 clusters.
We now run k-means with initialized centers (7,4),(1,3), and (5,9).

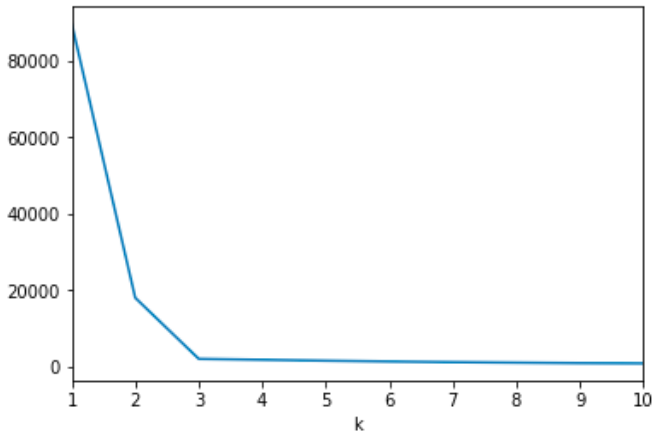
x	prob
(7,4)	-
(8,3)	
(5,9)	-
(3,3)	
(1,3)	-
(10,1)	

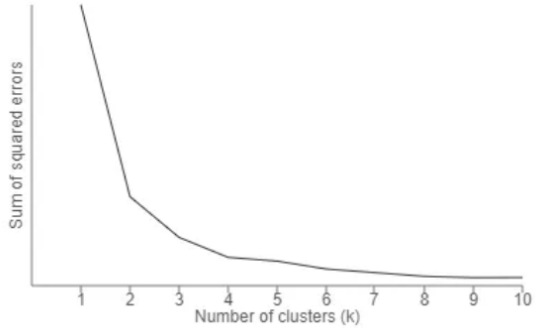


Issue 2: Choose k

- No principled way – no test set!
- Elbow method: see where you get saturation.
  - When WSS (Within-sum of squares) starts diminishing?

WSS





Ambiguous…

https://medium.com/analytics-vidhya/how-to-determine-the-optimal-k-for-k-means-708505d204eb