Explanatory Notes for 6.390

Shaunticlair Ruiz (Current TA)

Fall 2022

How to evaluate clustering algorithms

The biggest problem with clustering algorithms is that they're **unsupervised**: this makes it much harder to know if we've gotten a **good** result.

This is partly because our **loss** function doesn't necessarily tell us if clustering is **useful**, or represents the data **accurately**.

It just tells us if our points are **close** to their cluster **mean**. That doesn't always mean the clustering is **good**.

Example: Imagine **every** single point in the dataset gets its **own** cluster mean. The **distance** to the cluster mean would be 0 (low loss), but this isn't very **useful**!

This isn't useful because nothing has changed: we've gone from having n separate data points, to having... n separate clusters.

Clarification 1

The k**-means loss function** does **not** tell us if we have a good and **useful** clustering or not.

It only tells us if the points in our clusters are **close** to their **cluster mean**.

This can help us make **better** clusters, but that does not mean they are **good** or what we **want**.

Without having "true" labels, we have to find other ways to **verify** our approach.

We'll do two things to **approach** this problem:

- We'll look at some of the ways our **algorithm** can go wrong (or right).
- Then, we'll find **better** ways to evaluate our clusterings than just looking at the **loss**.

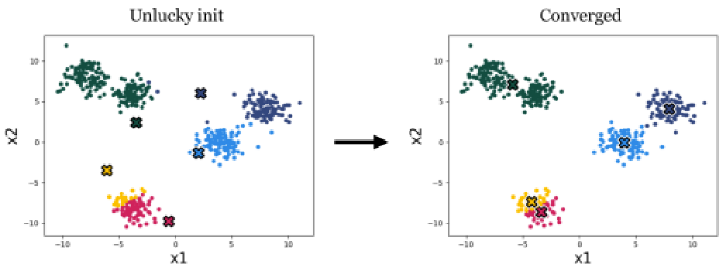
But, always remember that a "good" clustering is partly **subjective**, and depends on what you **want** to accomplish.

Initialization

The first problem we have is related to something we mentioned at the end of the last section: k-means is not **convex**.

That means we can find **local** minima that are not the **global** minimum: our **initialization** (our **starting** clusters) can affect whether we end up in a useful minimum.

The reason why is, mathematically, the same as when we first introduced the idea of a local minimum.



In this example, notice that we ended up with convergence on some very **bad** clusters: the bottom cluster is split in **half**!

The easiest way to resolve this is to run k-means multiple times with different initializations.

Other techniques exist, but this is the simplest one.

Concept 2

Getting an **unlucky initialization** can result in **clusters** that aren't **useful**.

We try to **solve** this by running our algorithm **multiple times**.

Choice of k

One important question we decided to **ignore** earlier was: **how many** clusters should we pick in advance?

Especially for **complex** data, we **don't know** how many natural clusters there will be.

But our number of clusters matter: because it's a parameter determines **how** our learning algorithm runs (rather than being chosen *by* the algorithm), it's a **hyperparameter**:

Concept 3

Our **number of clusters** k is a **hyperparameter**.

And, choosing too high *or* too low can both be **problematic**:

- If we set k too **high**, then we have more clusters than actually **exist**.
  - This can cause us to **split** real clusters in half, or find **patterns** that don't exist.
  - In a way, this resembles a kind of **overfitting**: we try to **closely** match the data, but end up fitting **too closely** and not **generalizing** well: **estimation error**.
  - **Example:** The **extreme** case looks like the example we mentioned **before**: when labeling animals, we could make... a different **species** for every single instance of **any** animal we find.

That doesn't sound very helpful.

- If we set k too **low**, we don't have **enough** clusters to represent our data.

- This means some clusters will be **lumped together** as a single thing: we **lose** some information.

- In this case, it's **impossible** to cluster everything in the way that would make the most **sense**: we have **structural error**.

- **Example:** Let's say we wanted to **sort** fish, birds, and mammals into **two** categories: we might just **divide** them into "flies" and "doesn't fly".

That's some information, but often not enough!

Concept 4

When choosing k (our **number of clusters**), we can cause **problems** by picking an inappropriate **value**:

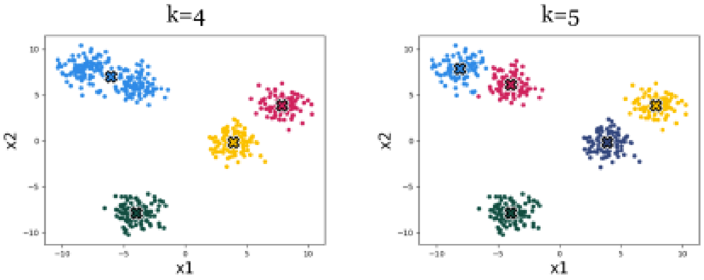
- **Too many** clusters (large k) can cause **overfitting** and **estimation error**: we find patterns we don't want.

- **Not enough** clusters (small k) causes **structural error**: it prevents us from correctly **separating** data.

Subjectivity of k

Not only is it hard to choose a "good" value of k, what a good value of k is can really depend on your opinion, and what you know about reality.

For example, consider the following example:



Which of these two clusterings is more accurate?

Should the top left be **one** cluster, or **two**? It's hard to say!

Even if you're **sure**, you might **disagree** with others, or find that the best one depends on your **needs**.

So not only can k values be too high or too low, they can also be **debatably** better or worse!

Concept 5

The **best** choice of **clustering** is not entirely objective: it can depend on your **opinion**, or how you plan to **use** the clustering.

What do we mean by, what we're "**using**" the clustering for? We'll get into that later, but in short: we might use **clusters** to make sense of **information**, or to make better **decisions**.

Different clusterings might be good when you want a different kind of understanding.

Example: The understanding you get from high-level comparisons (plants vs animals vs bacteria) is different from low-level comparison (cats vs dogs).