K-means Clustering

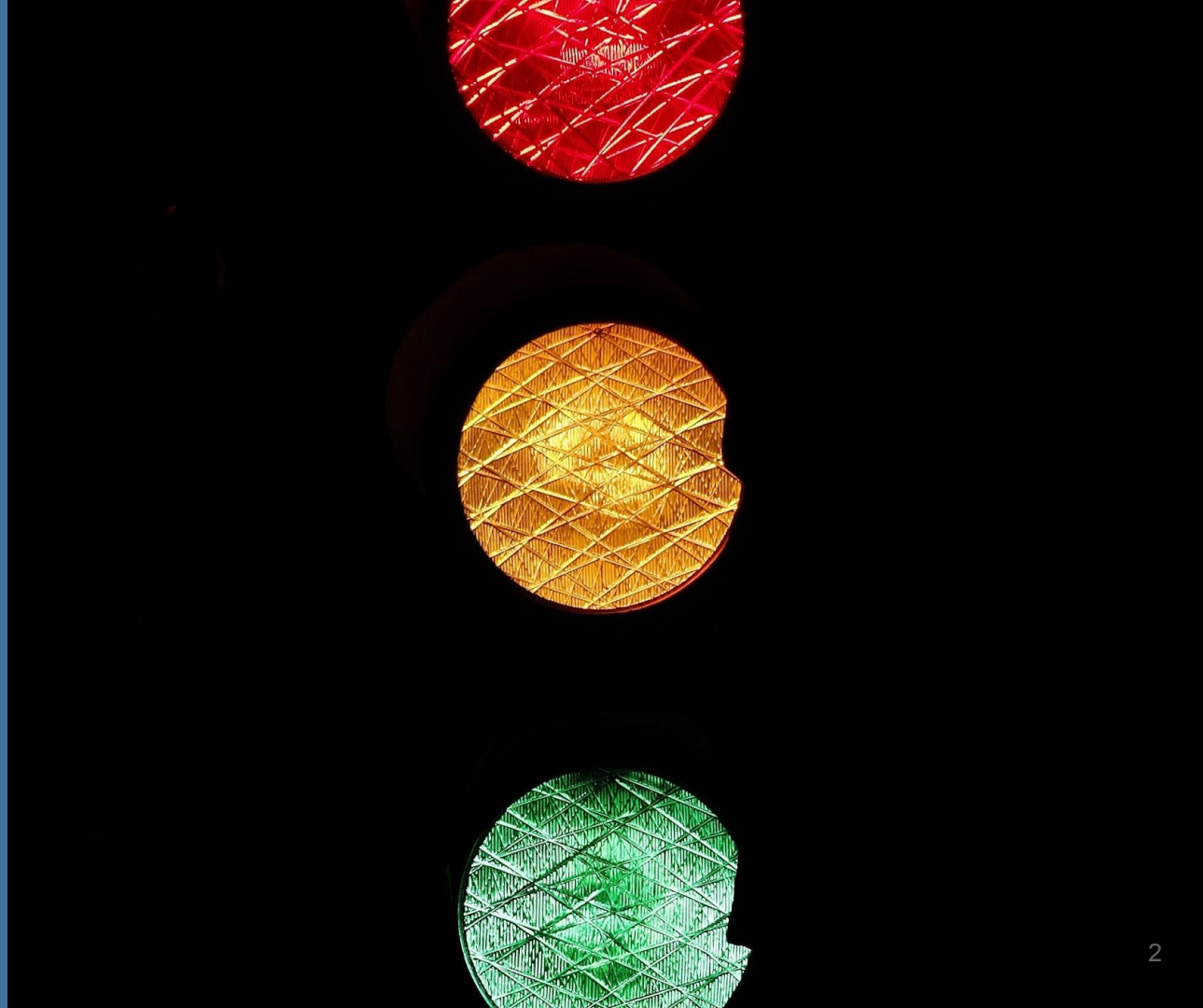
//Flatiron School

Review

Supervised vs Unsupervised Learning

Categorical vs Numeric

Target Variable vs Input Variables



By the end of the lesson students will be able to:

Assess what scenarios could use k-means

Articulate the methodology used by k-means

Apply `KMeans` from `sklearn.cluster` to a relevant dataset

Select the appropriate number of clusters using k-means and the elbow method

Evaluate the weaknesses and remedies to k-means

Lesson Preparation:

- Make sure you've pulled today's repository of lesson material
- Import the necessary packages
- Resource for code we will be using is here

In [3]:
```
# Required packages for today
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn import datasets

# Familiar packages for plotting, data manipulation, and numeric functions
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# Have plots appear in notebook
%matplotlib inline

# Default plot params
plt.style.use('seaborn')
cmap = 'tab10'
```

Scenario:

You work for the marketing department within large company that manages a customer base.

For each customer you have a record of average purchase cost and time since last purchase.

You **know** that if you want to retain your customers you cannot treat them the same. You can use targeted marketing ads towards groups that demonstrate different behavior, but how will you divide the customers into groups?

Scenario Review:

Target:
Groups (or Categories)

Problem type:
Unsupervised

You work for the marketing department within large company that manages a customer base.

For each customer you have a record of average purchase cost and time since last purchase.

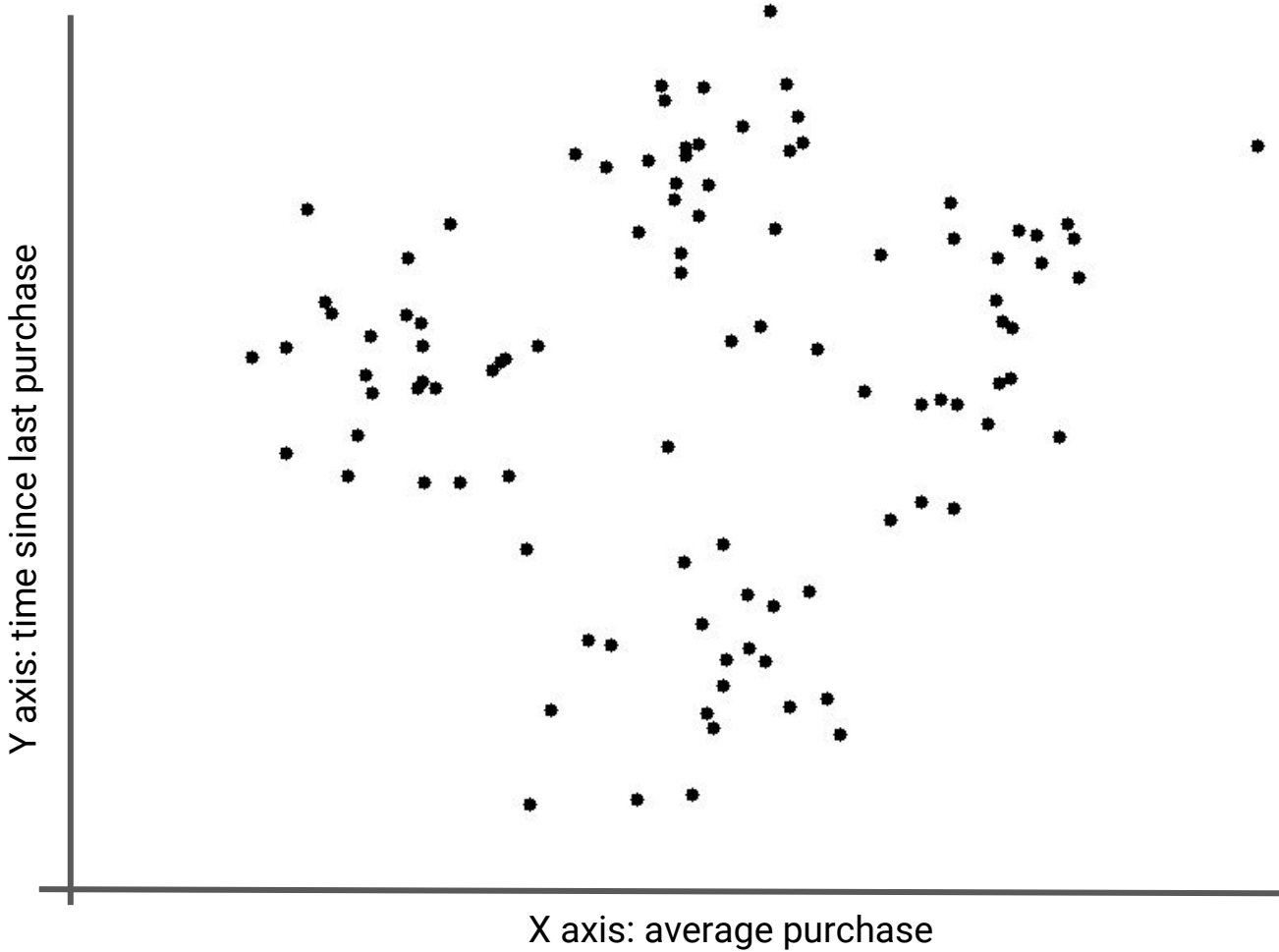
You **know** that if you want to retain your customers you cannot treat them the same. You can use targeted marketing ads towards groups that demonstrate different behavior, but how will you divide the customers into groups?

Scenario:

Without the aid of an algorithm, how would you separate them into groups?

How many groups?

Discuss with your neighbor.



Observe four different methods:

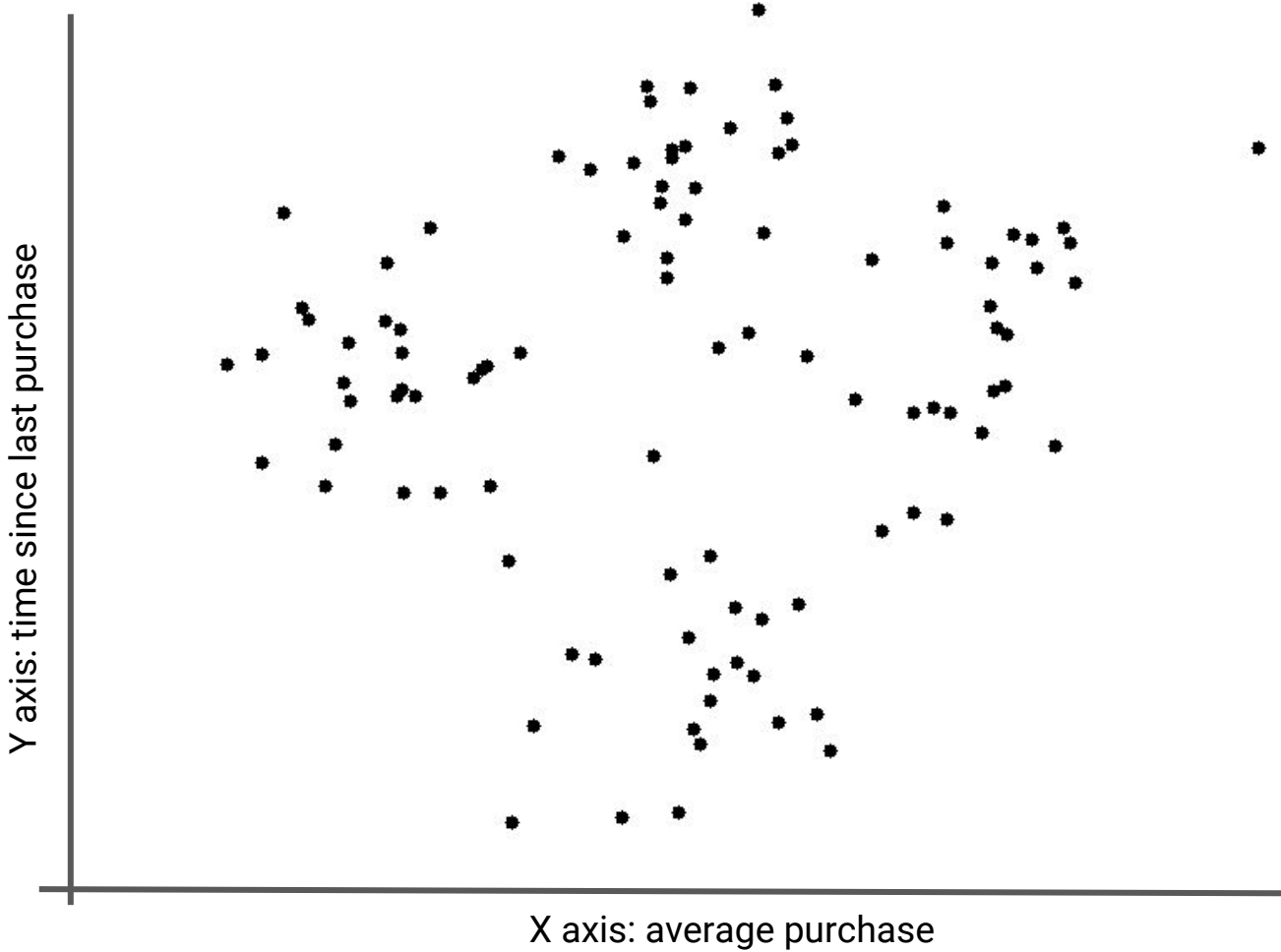
What do they have in common?

What differences are there between them?

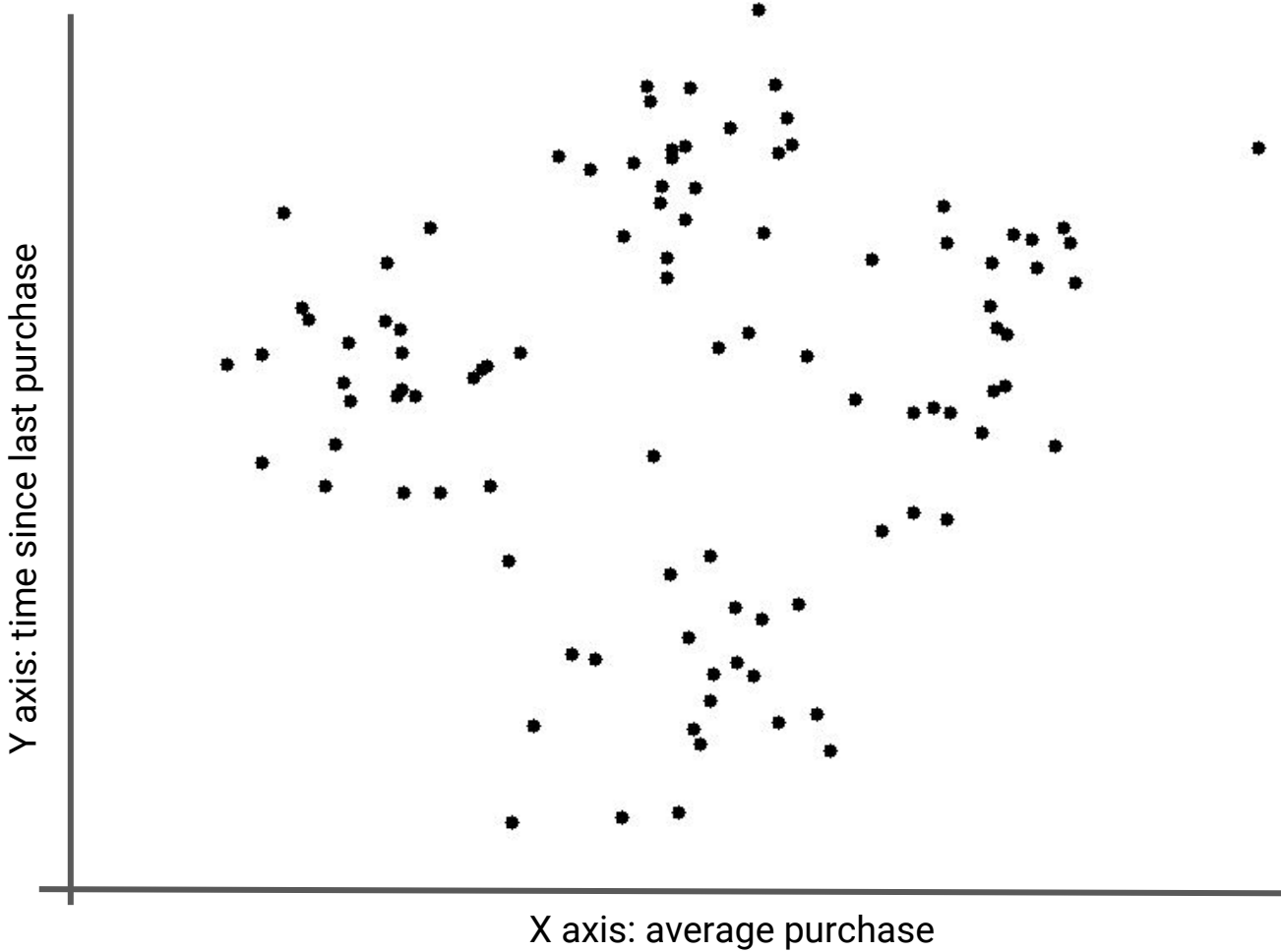
How many groups are there at the end?

Do you see any problems with these methods?

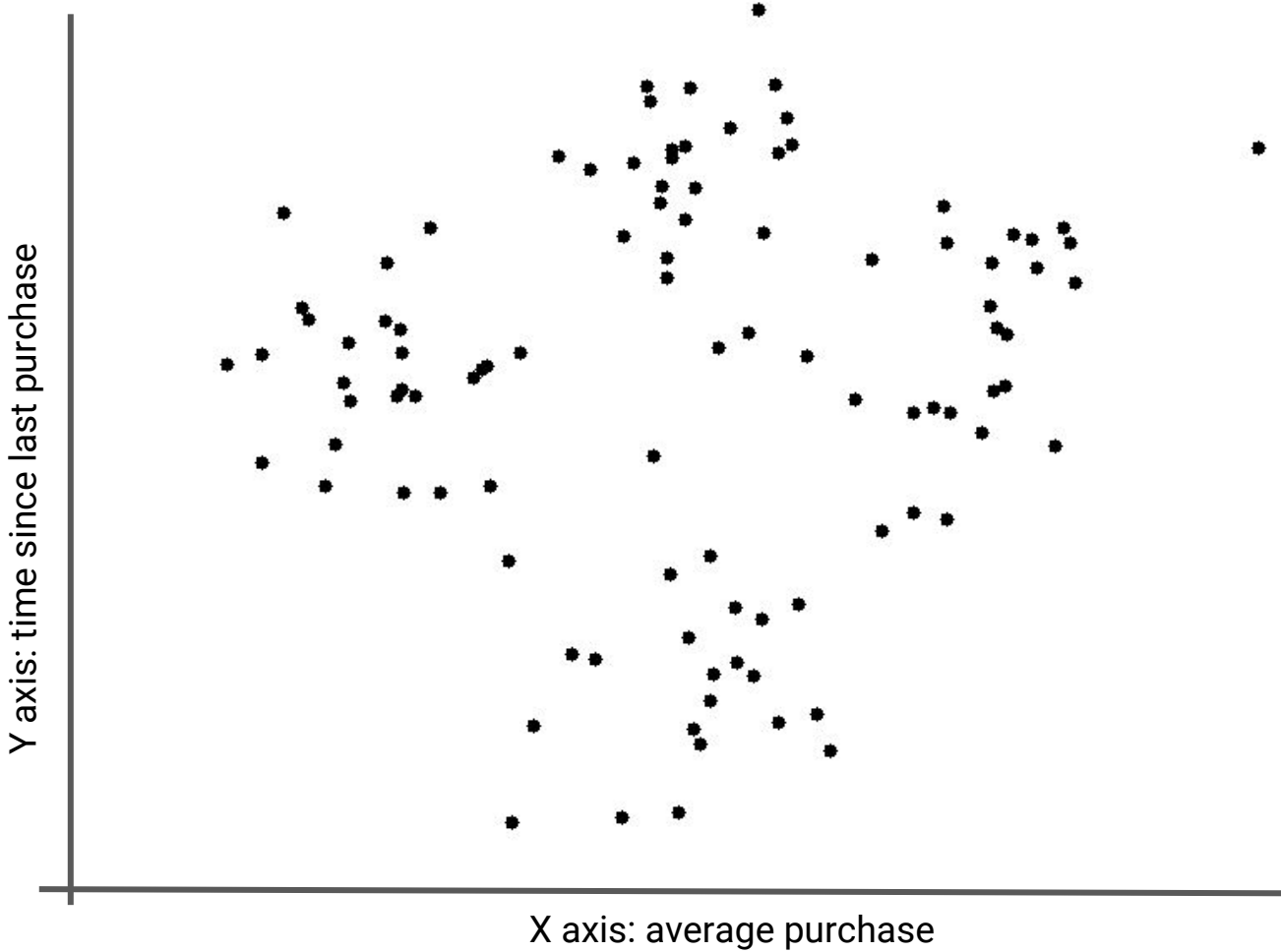
Method 1:



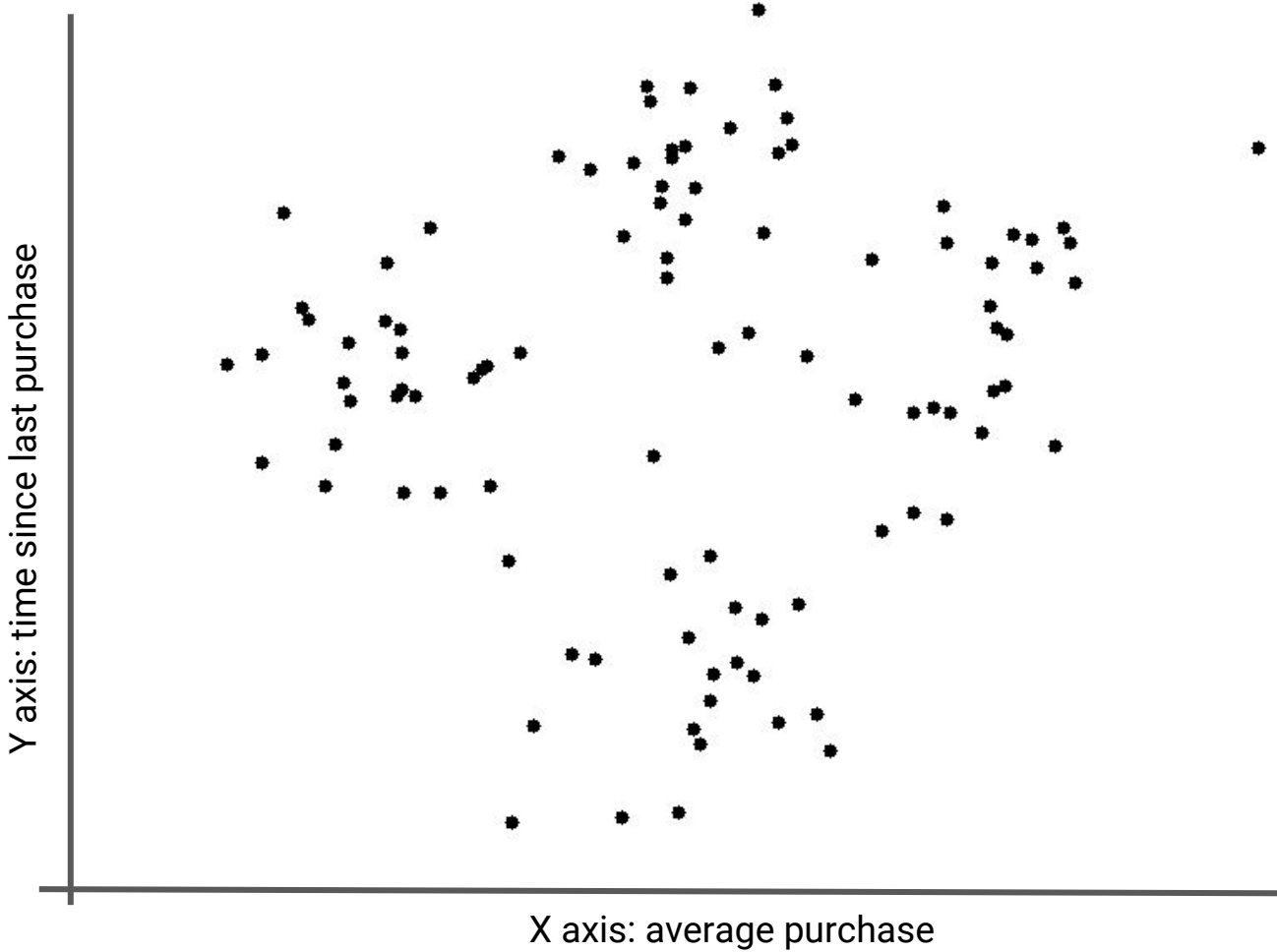
Method 2:



Method 3:



Method 4:



Review four different methods:

What do they have in common?

- **Green dots starts at points**
- **Calculates distance**
- **Moves dots**
- **Re-measures distance**
- **Moves dots as needed**

What differences are there between them?

Dots start in different places and groups settle in different places

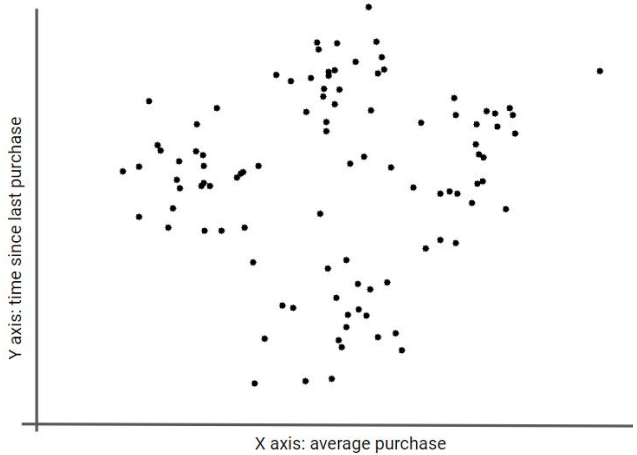
How many groups are there at the end?

Four

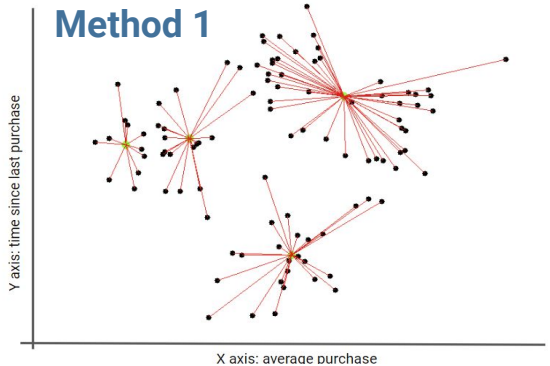
Do you see any problems with these methods?

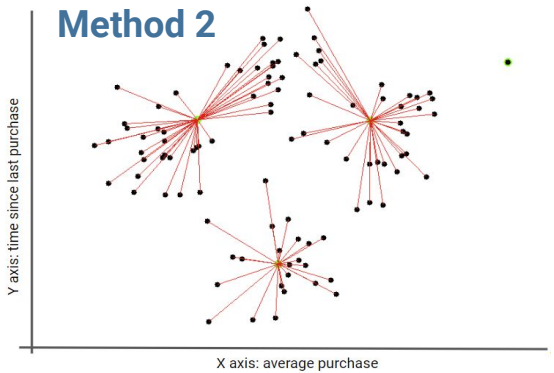
Too variable

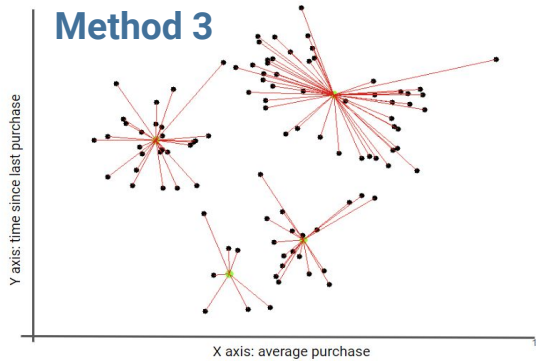
Four groups (k=4) produces unreliable results

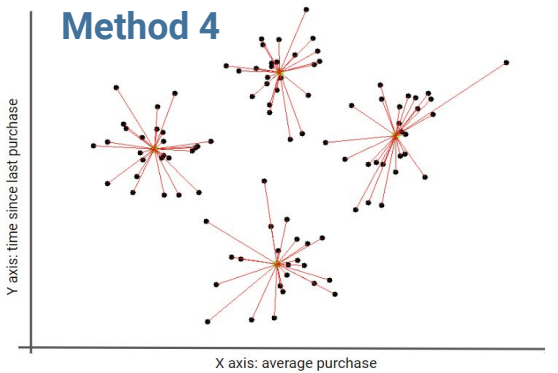


K in “k-means” refers to how many groups you specify









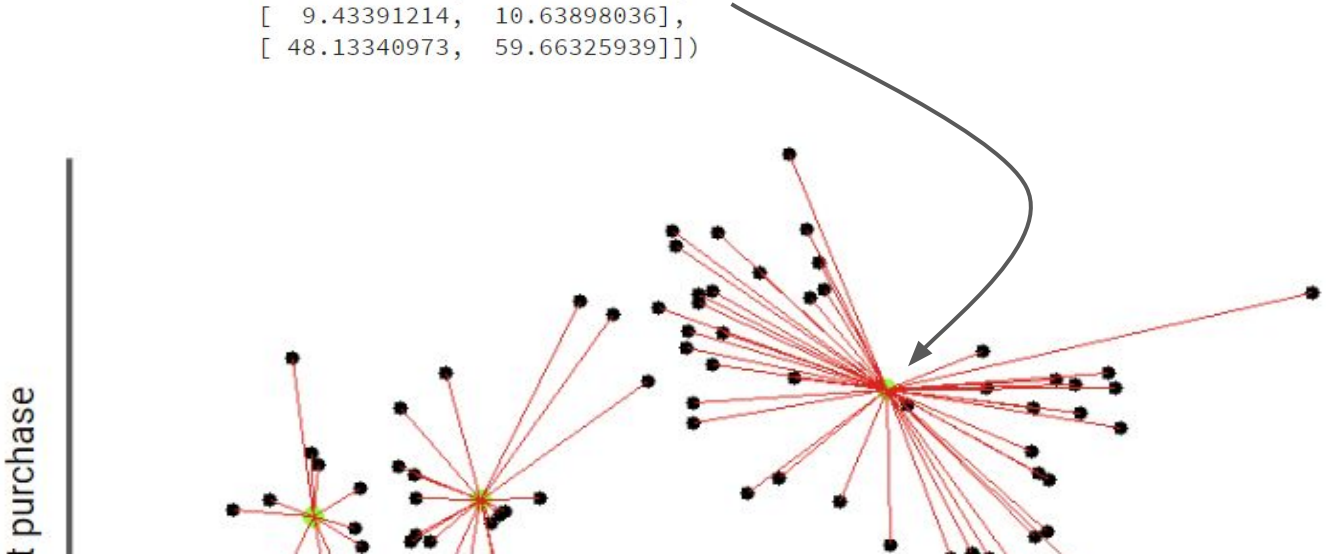
Process:

1. Pick and specify k (n_clusters)
2. Algorithm return an array of “cluster centers” or “centroids”

```
In [20]: model = KMeans(n_clusters=4).fit(test)

In [21]: model.cluster_centers_

Out[21]: array([[ 69.92418447, -10.11964119],
                [ 31.78831061,  59.67227949],
                [  9.43391214,  10.63898036],
                [ 48.13340973,  59.66325939]])
```



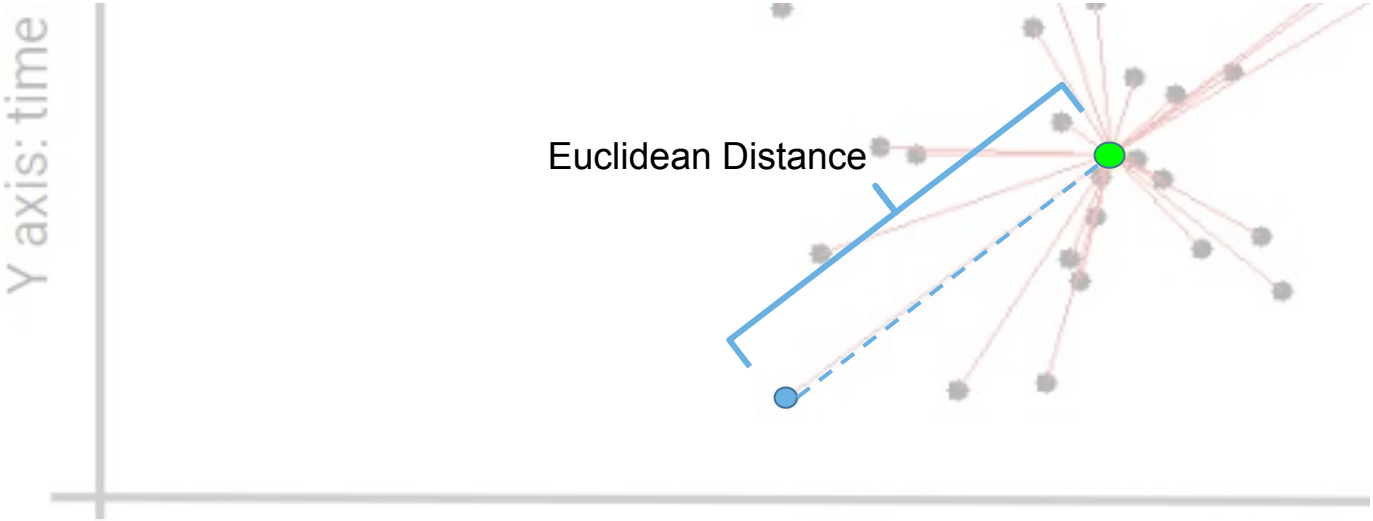
Alert! New vocabulary!

‘K’ is the number of **clusters**, or groups, in the dataset you specify the algorithm define.

The **centroids**, or cluster centers, are the points at the **center** of each **cluster**. The coordinates of the centroid are the **mean value** of each variable within the defined group.

K-means is called such because it returns the **mean** values of **k**-specified **clusters**.

K-means uses Euclidean Distance

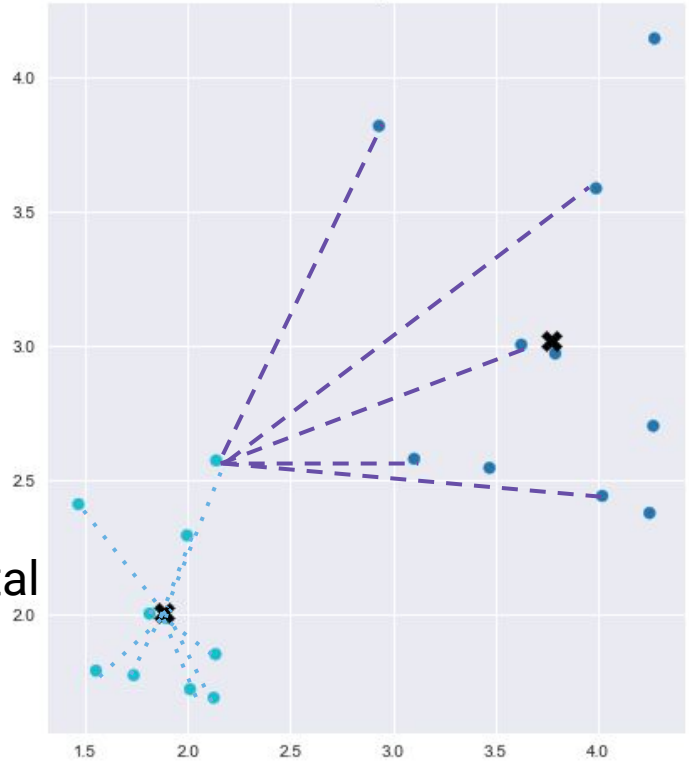


$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

K-means is an optimization algorithm

That recalculates centroids and reassigns group labels in order to:



MAXIMIZE sum total of inter-cluster distance

MINIMIZE sum total of intra-cluster distance

Silhouette coefficient and elbow method

Silhouette coefficient ranges between 1 and -1. The closer to 1 the more clearly defined are the clusters. The closer to -1, the more incorrect assignment.

It calculates a score for all data points and then averages across all points.

Elbow method uses the sum of squared error (`inertia_` in k-means python) for all points

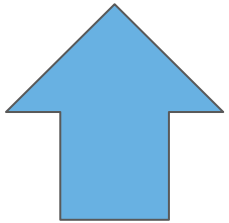
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

A refers to the average distance between a point and all other points in that cluster.

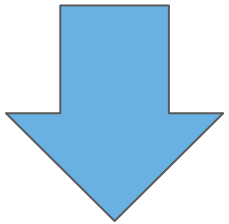
B refers to the average distance between that same point and all other points in clusters to which it does not belong

Silhouette coefficient and elbow method

Silhouette coefficient - want K with

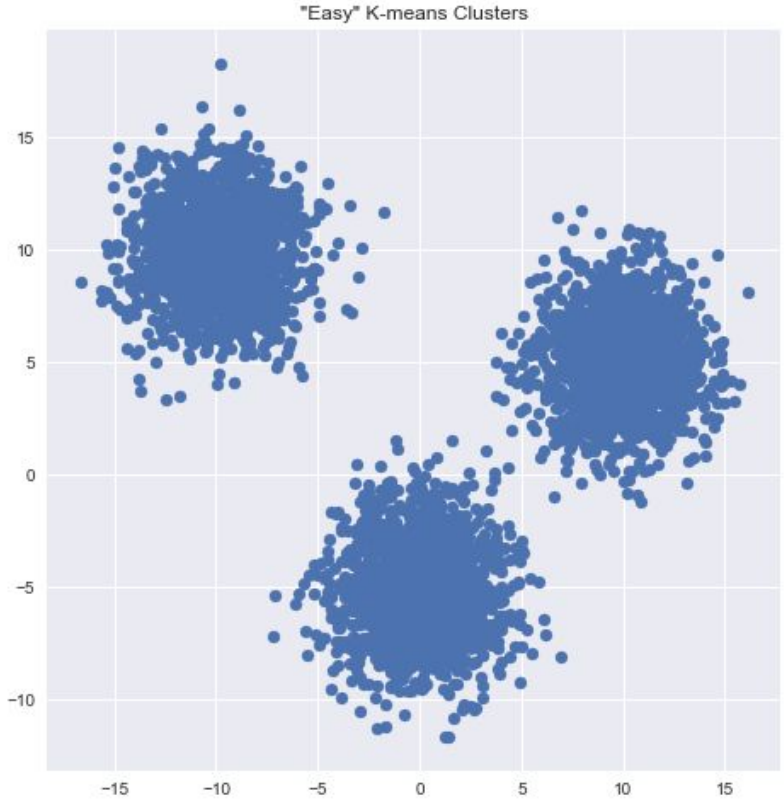


Elbow method - want k with



Find best k

Ideal K-means scenario



Assumptions of K-means

- Independent variables
- Balanced cluster sizes
- Clusters have similar density
- Spherical clusters/equal variance of variables

When K-means struggles