Introduction to Clustering

Learning Objectives

- Understand what we mean by clustering
- Format and preprocess data for clustering
- Perform a K-Means clustering analysis
- Evaluate clusters for fit

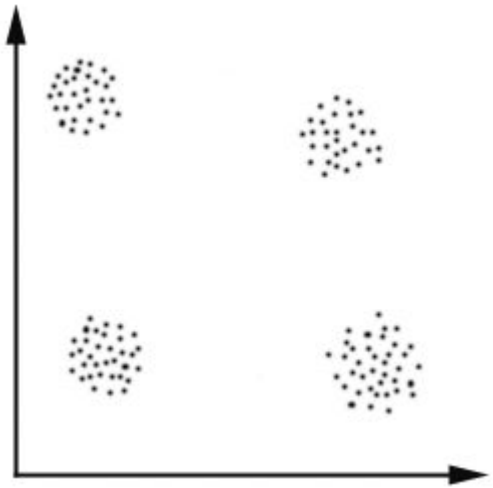
Clustering

- **Classification** - create a model to predict which group a point belongs to
- **Clustering** - find groups that exist in the data already

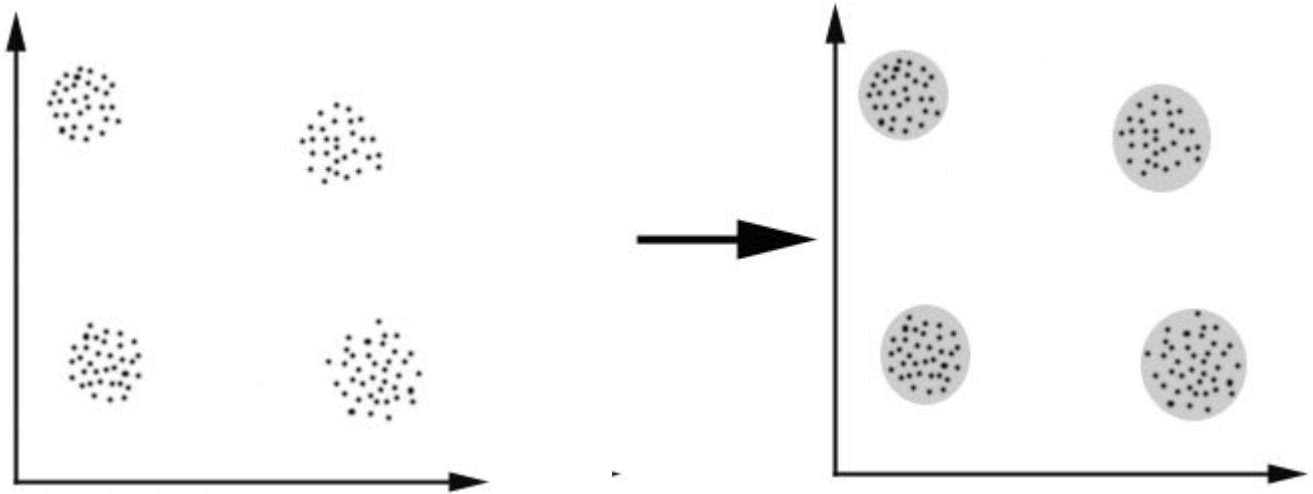
Helpful uses for clustering:

- Find items with similar behavior (users, products, voters, etc)
- Market segmentation
- Understand complex systems
- Discover meaningful categories for your data
- Reduce the number of classes by grouping (e.g. bourbons, scotches -> whiskeys)
- Reduce the dimensions of your problem
- Pre-processing! Create labels for supervised learning

Algorithm clustering

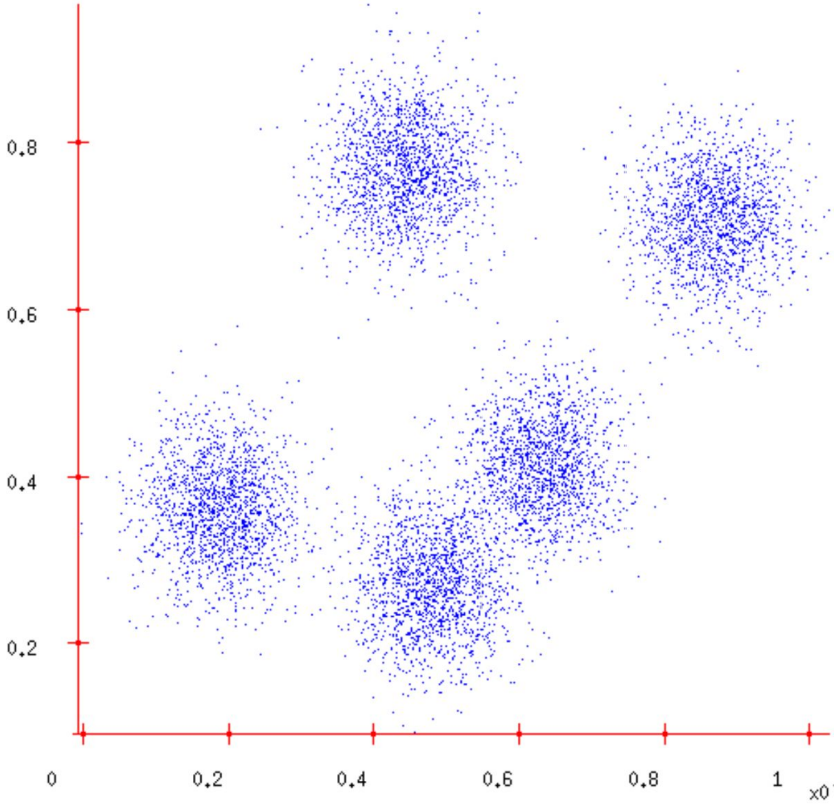


Algorithm clustering



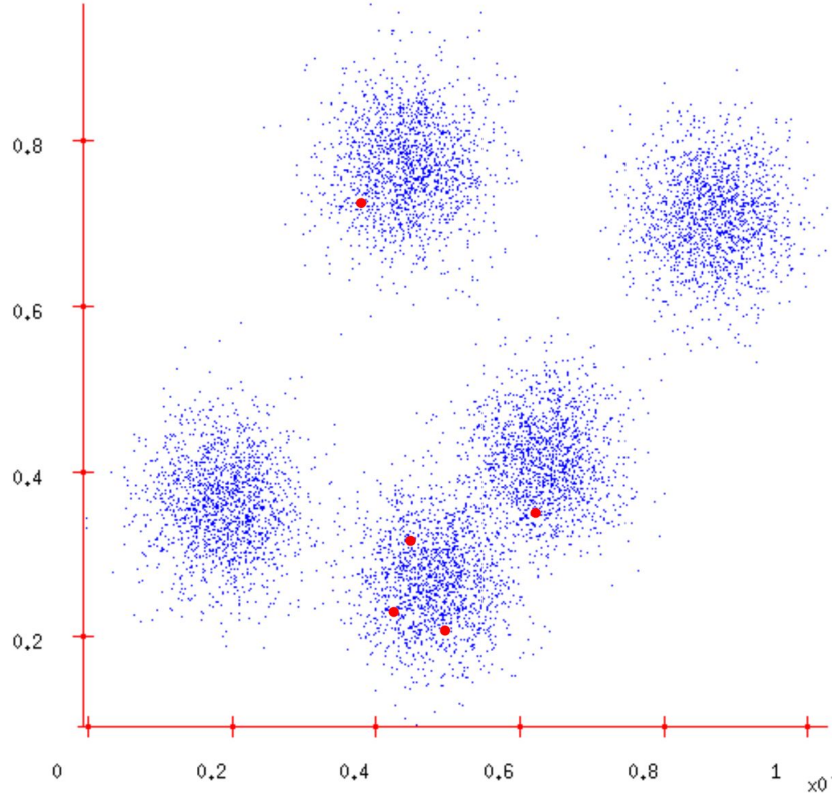
Algorithm clustering

1. **Pick a value for k (the number of clusters to create)**



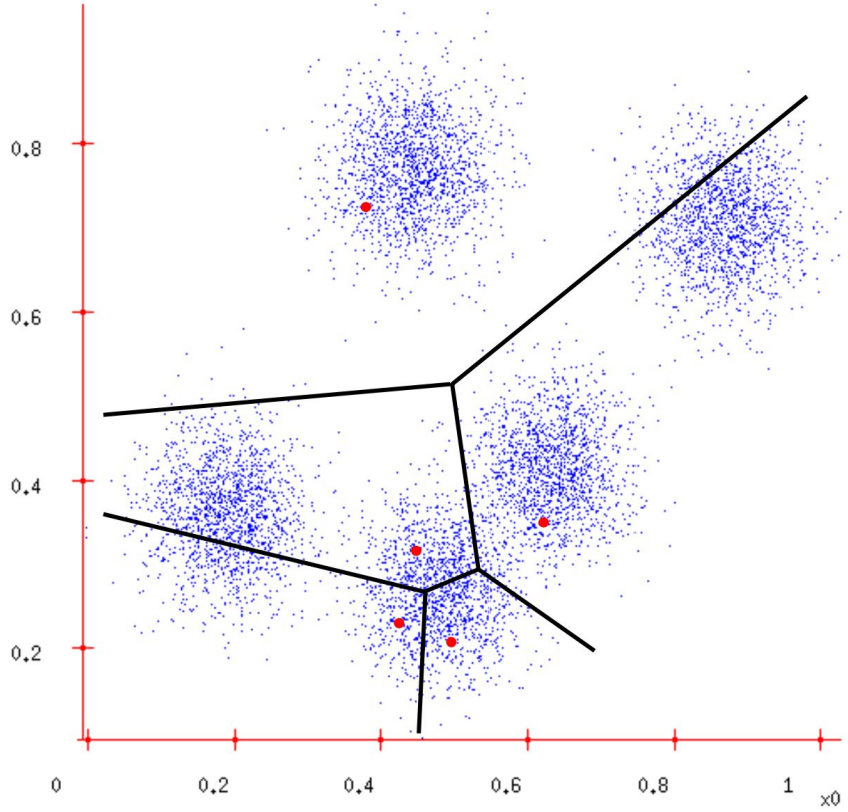
Algorithm clustering

2. **Initialize k 'centroids' (starting points) in your data**



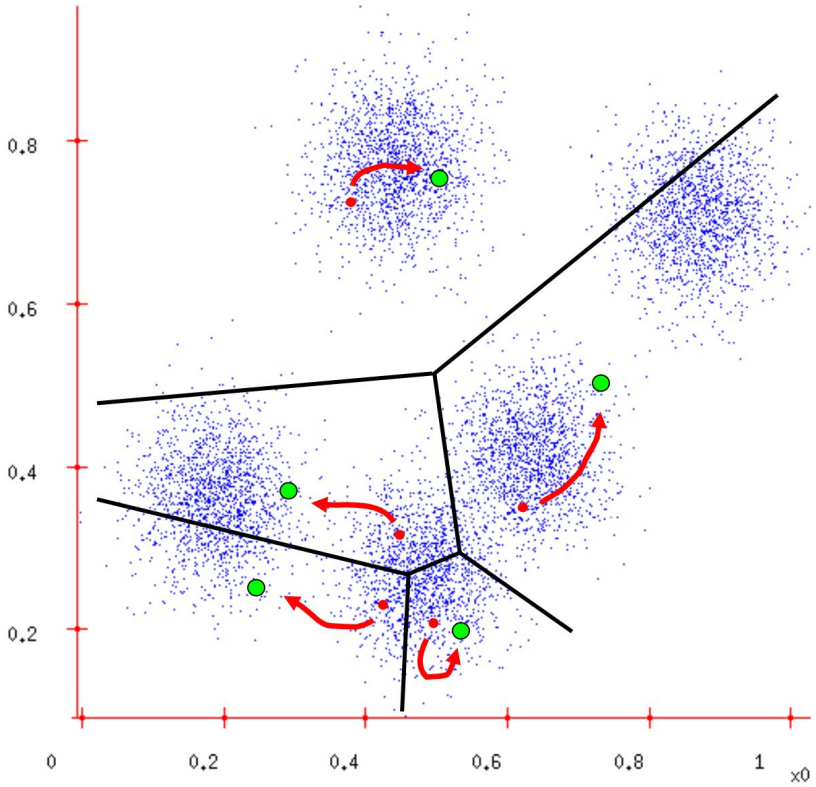
Algorithm clustering

3. Assign each point to the nearest centroid. These are your 'clusters'.



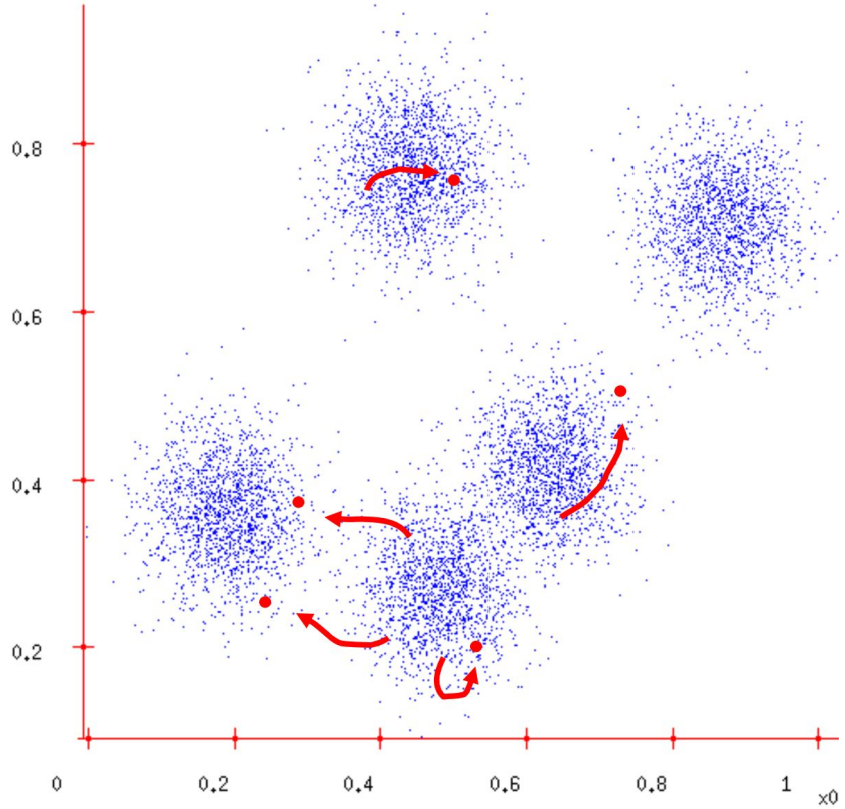
Algorithm clustering

4. Make your clusters better. Move each centroid to the center of its cluster.

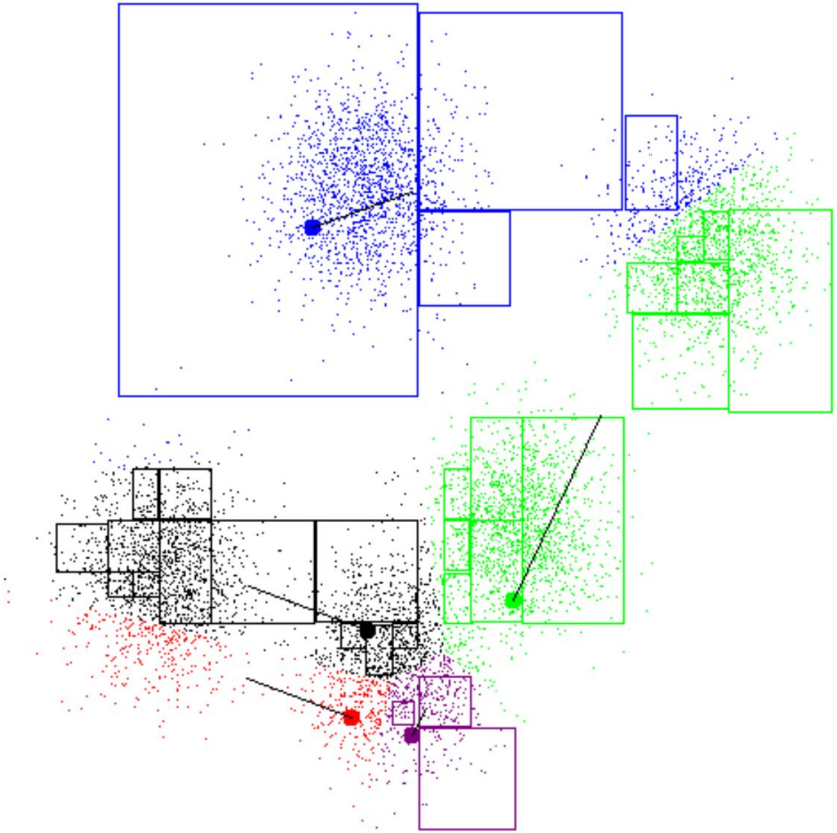


Algorithm clustering

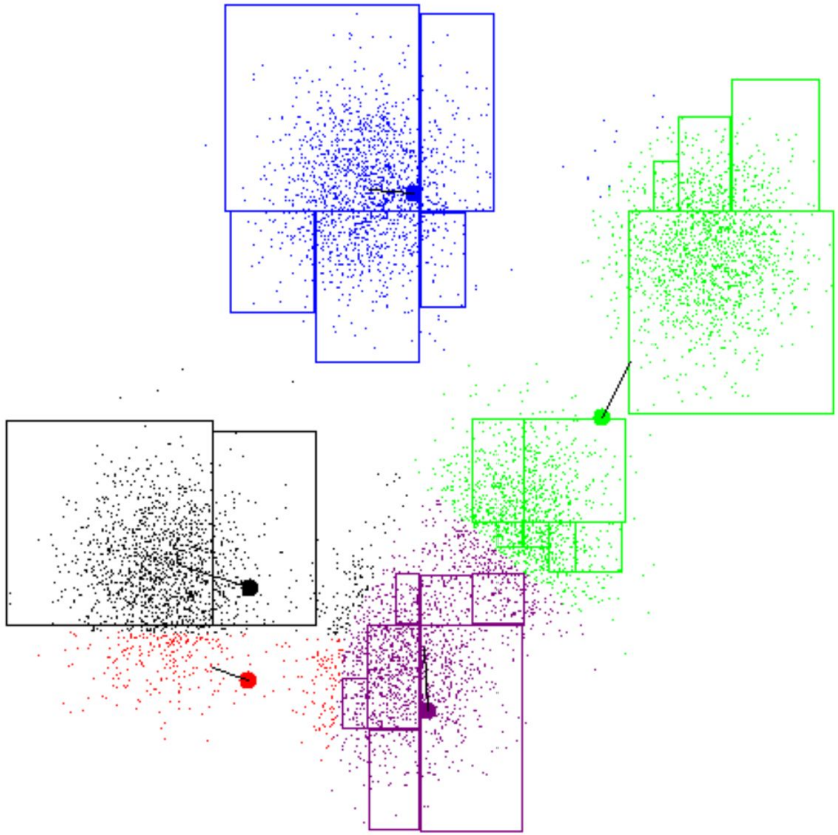
5. **Repeat steps 3-4 until your centroids converge.**



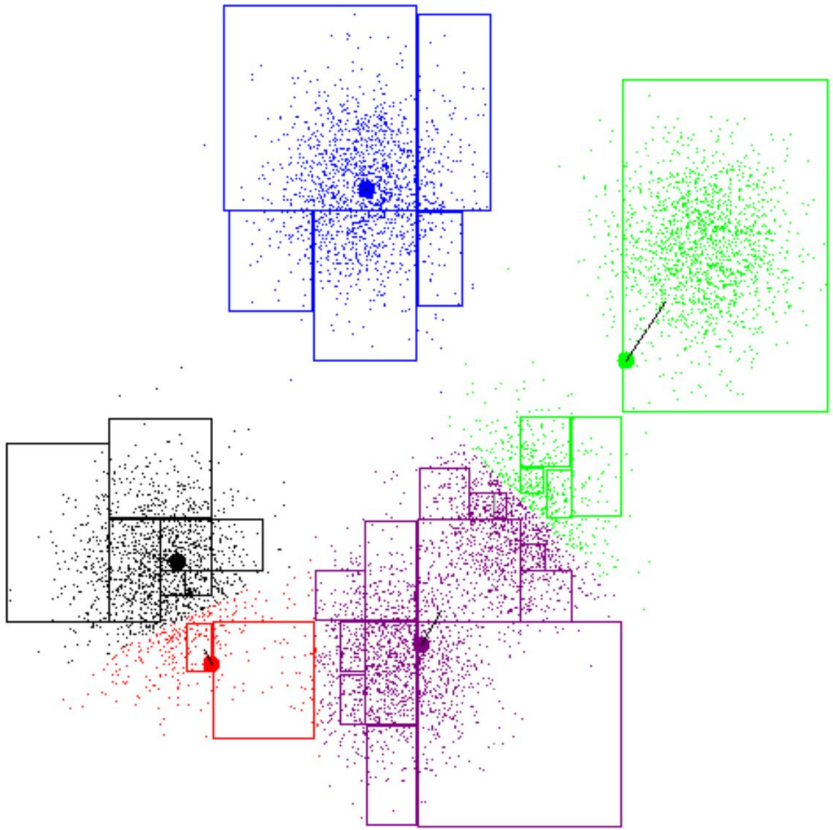
Algorithm clustering



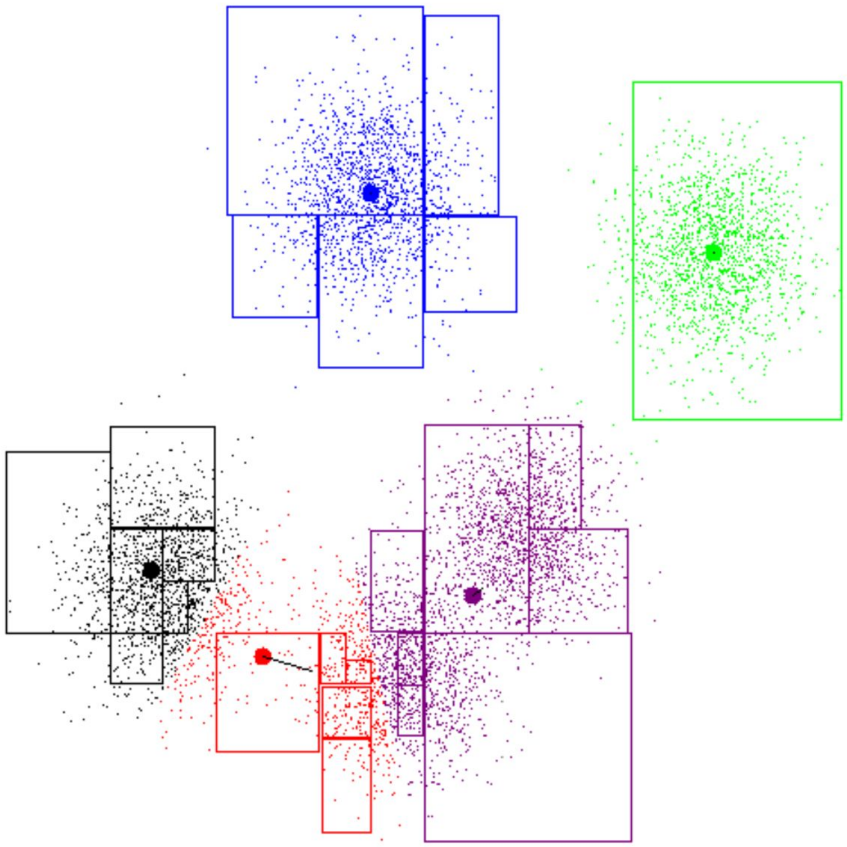
Algorithm clustering



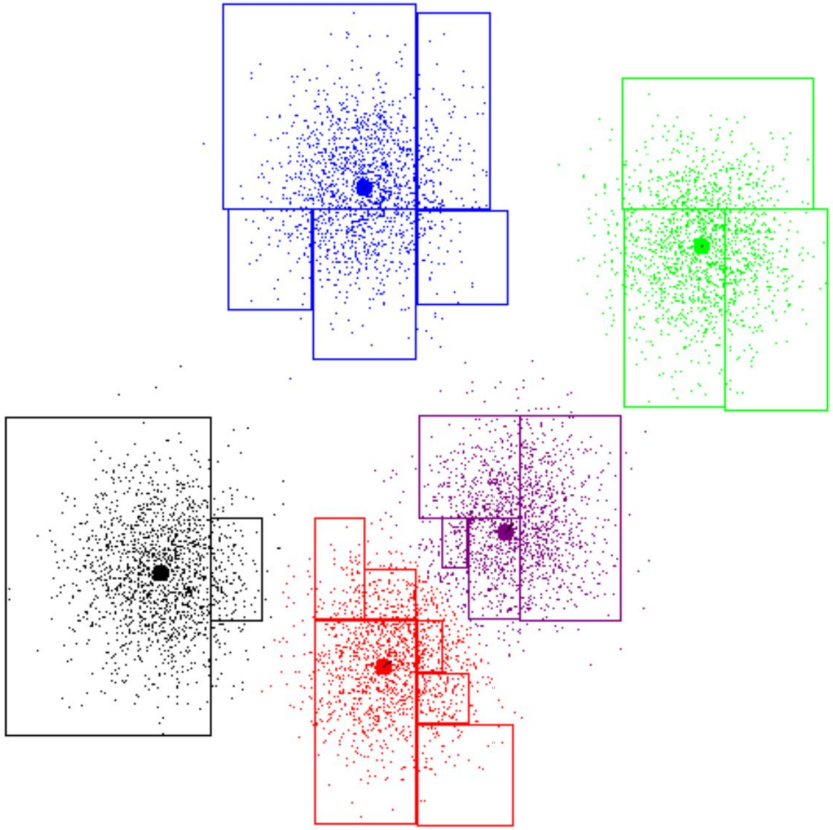
Algorithm clustering



Algorithm clustering



Algorithm clustering



Metrics for assessing clusters

Inertia

- Measures how close the elements of each clusters are to the centroid
- Compute the sum of squared error for each cluster
- Ranges from 0 to very high values
- Low inertia == dense clusters
- **Lower is better!**

$$\sum_{j=0}^{n} (x_j - \mu_i)^2$$

Metrics for assessing clusters

Silhouette Score

- Measures how far apart clusters are.
- how much closer data points are to their own clusters than to the nearest neighbor cluster.
- Ranges from -1 to 1
- High silhouette score == clusters well separated.
- **Higher is better!**

Clustering using `scikit-learn`

- **Select the number k of clusters**
- **Scale your data to normalize the distances in the data**

1. **MinMaxScaler()**

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(df[cols])
X_scaled = scaler.transform(df[cols])
```

2. **StandardScaler()**

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(df[cols])
X_scaled = scaler.transform(df[cols])
```

Clustering using `scikit-learn`

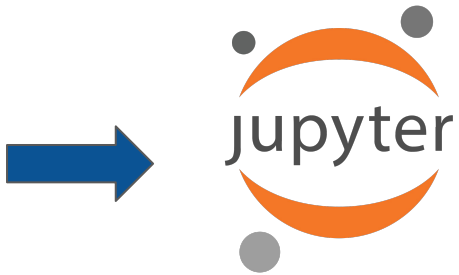
```
# import KMeans class from sklearn
from sklearn.cluster import KMeans
# specify the number of k
k = 2

# Initialize model
kmeans = KMeans(n_clusters=k, random_state=42)

# fit the data
kmeans.fit(X_scaled)
```

Clustering using `scikit-learn`

How can we determine the labels, centroids and evaluate the inertia?

- **Labels:** By calling the ***.labels_*** of the model
- **Centroids:** By calling the ***.cluster_centers_*** **of model**
- **Inertia:** By calling the ***.inertia_*** of the model

How do you determine the silhouette score?



Working with API

Working with API

Learning objectives

- Identify relevant HTTP Verbs & their uses.
- Describe Application Programming Interfaces (APIs) and know how to make calls and consume API data.
- Access public APIs and get information back.
- Read and write data in JSON format.
- Use the **requests** library.

What is API?

- Stands for Application Programming Interface
- a set of routines, protocols, and tools for building software applications.
- It specifies how software components should interact.
- For example, the **plot()** call of Matplotlib is an API. We do not care how it works.

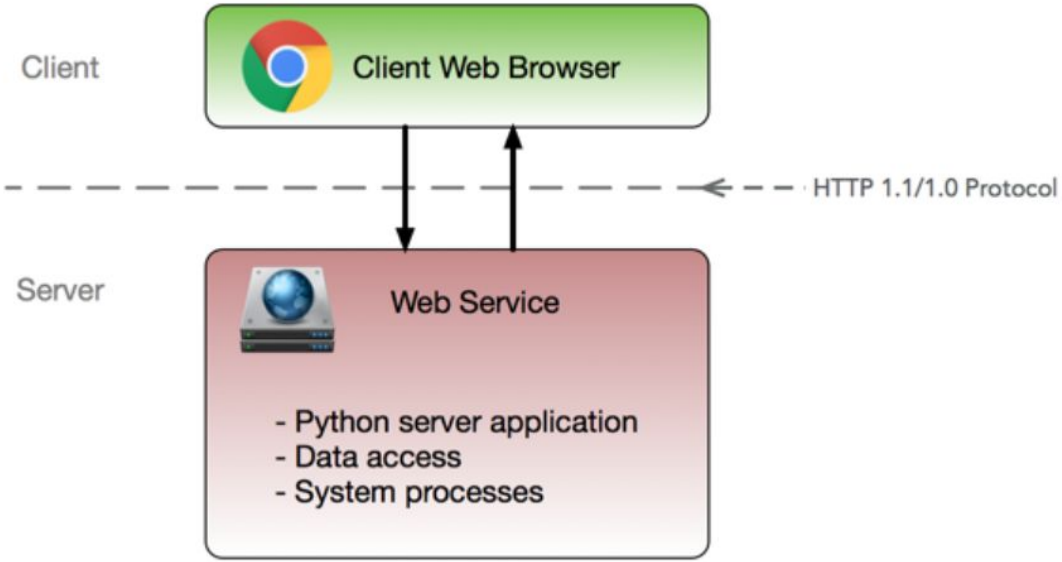
API Tokens, HTTP, clients and Servers

- **API Tokens**
  - Many APIs are free to access
  - But registration is required (as a developer) and authorization key generated
  -
- **HTTP**
  - **HyperText Transfer Protocol:** System of rules **(Protocols)** that determine how web pages **(HyperText)** get sent **(Transfer)**.
  - Determined format of message between **HTTP Clients** and **HTTP servers**

API Tokens, HTTP, clients and Servers

- **HTTP clients**
  - These are what make or generate **HTTP requests**
  - Examples include browsers, CLI, application code such as **Python requests**
  -
- **HTTP Servers (web servers)**
  - Receive HTTP requests, pass them to the necessary location and generate responses
  - They house the **web applications** that process the HTTP requests

API Tokens, HTTP, clients and Servers



HTTP Requests and Responses

Request URL: https://generalassemb.ly/

Request Method: GET

Status Code: 200 OK

Remote Address: 52.4.202.19:443

Referrer Policy: no-referrer-when-downgrade

▼ **Request Headers** view parsed

```
GET / HTTP/1.1
Host: generalassemb.ly
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x
64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
80.0.3987.163 Safari/537.36
Sec-Fetch-Dest: document
Accept: text/html,application/xhtml+xml,applicatio
```

HTTP Request methods

- **GET** => Retrieve a resource.
- **POST** => Create a resource.
- **PATCH** (*or **PUT**, but **PATCH** is recommended*) => Update an existing resource.
- **DELETE** => Delete a resource.
- **HEAD** => Retrieve the headers for a resource.

HTTP Responses

Request URL: https://generalassemb.ly/

Request Method: GET

Status Code: 200 OK

Remote Address: 52.4.202.19:443

Referrer Policy: no-referrer-when-downgrade

Response Headers view parsed

```
HTTP/1.1 200 OK
Connection: keep-alive
Server: nginx
Date: Sun, 26 Apr 2020 07:50:57 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Status: 200 OK
```

HTTP Responses Status Codes

Code	Reason
200	OK
301	Moved Permanently
302	Moved Temporarily
307	Temporary Redirect
400	Bad Request
403	Forbidden
404	Not Found
500	Internal Server Error

JSON

- Short for JavaScript Object Notation
- A collection of Name/Value pairs (dictionary)
- And ordered list of lists

```
{"widget": {
    "debug": "on",
    "window": {
        "title": "Sample Konfabulator Widget",
        "name": "main_window",
        "width": 500,
        "height": 500
    },
    "image": {
        "src": "Images/Sun.png",
        "name": "sun1",
        "hOffset": 250,
        "vOffset": 250,
        "alignment": "center"
    },
    "text": {
        "data": "Click Here",
        "size": 36,
        "style": "bold",
        "name": "text1",
        "hOffset": 250,
        "vOffset": 100,
        "alignment": "center",
        "onMouseUp": "sun1.opacity = (sun1.opacity / 100) * 90;"
    }
}}
```

Any Questions/ Session attendance log (module 4 day 1)