K-Mean Clustering

M T W T F S DATE:

- Section Overview
  - understanding clustering
  - History and Intuition of K-mean
  - Mathematical Theory of K-Means
  - Example of K-mean
  - K-mean project Exercise.

Clustering General Overview

Clustering uses unlabelled data and looks for similarities between groups (clusters) in order to segment the data into seperate clusters.

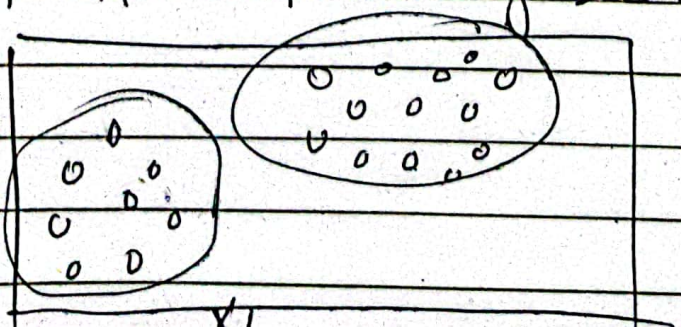
- Keep in mind that we don't actually know the true correct label for this data.

Imagine an example dataset

X1	X2
2	4
6	3
5	4
2	1

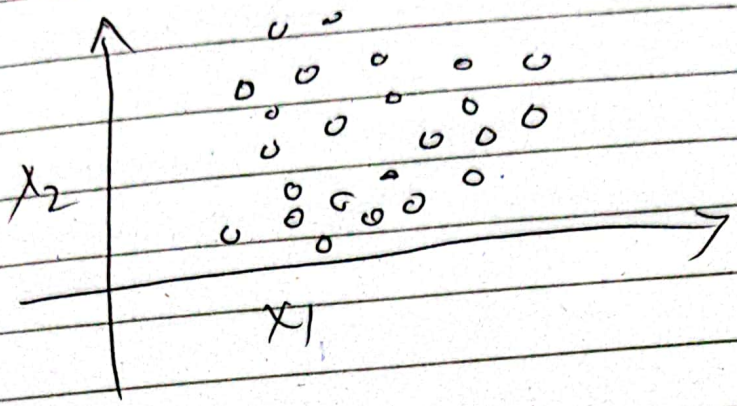
- distance is the intuitive metrics

If we plot it intuitively see 2 groups



MIDIDES DATE:

- 2 or 3 clusters could be reasonable.



Different methods can be used to decide! number of clusters.

- Some methods themself decide and for some we need to decide the number of clusters.

methods (will name ~~as~~ group 0, group 1, group 2

Main clustering Idea:-

- Use features to decide which points are most similar to other points.
- Realize the there are no ~~$~~ final correct y label to compare cluster results to.
- We can think of clustering as an unsupervised learning process that discovers potential labels.

DATE:

M T W T F S

Unsupervised paradigm shift

How we assign a new datapoint to a cluster?

- Different approches depending on the unsupervised learning algorithm used-
- Use features to assign most appropriate cluster-
- just as before, there is no way to measure if this was the "correct" assignment

- If we have discovered these new clusters labels, could we use that as a y for supervised training?
  - Yes we can use unsupervised learning to discover possible labels, then apply supervised learning on new data points.

What is trade off?

- Clustering doesn't tell you what these new cluster labels represent, no real way of knowing if these clusters are truely significant.

- Clustering ideas still to come:
  - How do we decide which number of cluster is based best?
  - How we decide or let the algorithm decide-
  - How can we measure "goodness of fit" for clustering without a y label for comparison-

DATE:

(M)(I)(B)(I)(F)(S)

- Machine learning as an art:
  - What is ground truth
  - What tradeoffs are we making by using unsupervised learning as a substitute for ground truth of they label that was not given?

K-mean Clustering

History, Intuition and Theory

- Hugo steinhaus was the mathematician who began laying out groundwork for clustering methods.

- 1940s - 1950s : Hugo steinhaus polish mathematician and professor.
- During WWII he was in hidding from nazi occupation, yet still taught clandestine math classes from memory, since higher education for poles was forbidden.

- He even established statistical method for estimating Germen casualty rates based ~~on~~ simply based on relative frequencies of obituaries in local newspapers.
- After WWII he helped established the Mathematics department at University

M T W T F S DATE:

Wroclaw -

- It was at the University of Wroclaw where he began to develop the idea that would become the foundation of K-mean clustering!
- check out his wikipedia page for more interesting stories from his incredible life.

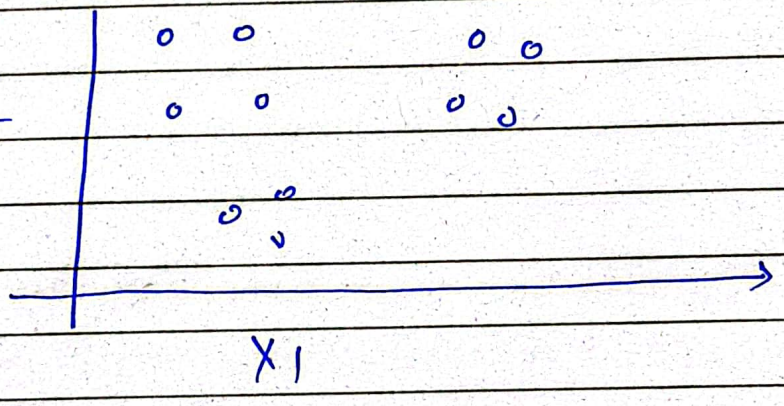
- 1957 : Stuart Lloyd of Bell Labs developed the standard algorithmic technique for clustering related to K-Means.
- However Lloyd did not publish his technique in a journel until 1982.
- 1965 Edward W. Forgy published the clustering techniques, thus both Lloyd and Forgy are credited with the development.

- 1967 : James MacQueen of UCLA develops and published "some methods for classification and analysis of multivariate observation".
- This is the first publisheaction to introduce the term "K-Mean" as a sequential clustering algorithm

MIDIDS DATE:

- First a set of properties point must satisfy:
  - Each point must belong to a cluster.
  - Each point can only belong to one cluster (no single point can belong to multiple clusters)

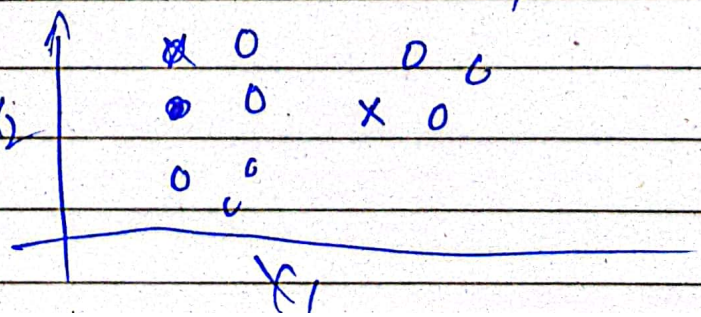
How to kmean cluster works



- step 1: chose the number of cluster to create (this is K value)

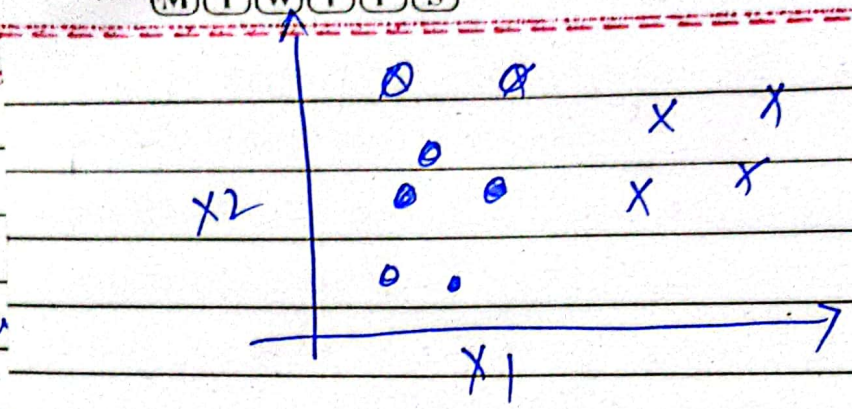
  : We have choose K=3. Note in most situation you wount visualize the data.

step 2: Random select K distinct data points. Our K = 3;



Step 2: Assign each ~~point~~ new K points we feel these our "cluster points".

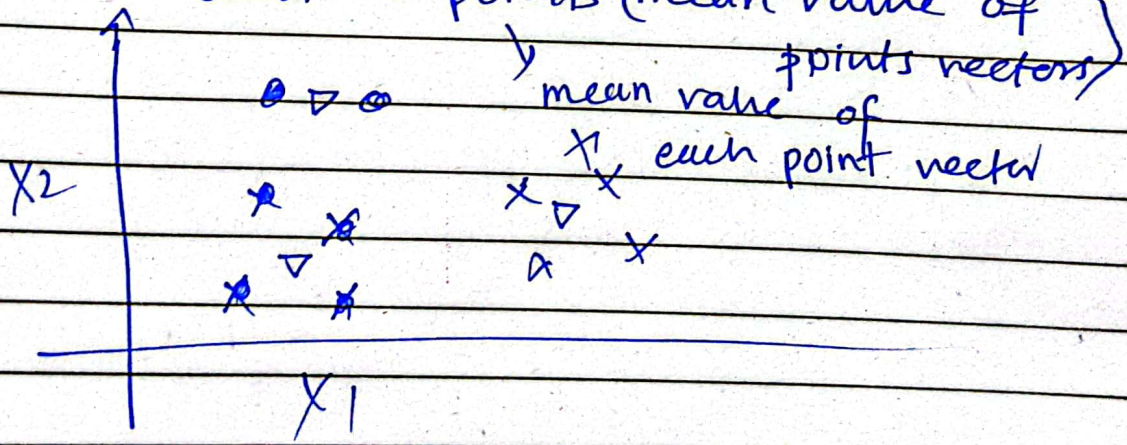
M T W T F S DATE:



Step 3: Assign each remaining points to the 'nearest "cluster" point)

- Note this is using a distance metric to judge the nearest points

Step 4: Calculate the center of the clusters points (mean value of points vectors)



Step 5: Now assign each point to the nearest cluster center.

6. - We repeat step 4 and 5 until there are no more cluster reassignments

NOTES DATE:

Upcoming considerations:

- How do we choose a reasonable value for k number of clusters?
- Is there any way we can evaluate how good our current k value is at determining clusters?

K-mean clustering coding # part 1

```
df = pd.read_csv("../Data/Bank-full.csv")
df.info()
df.describe()

# two things - domain experience
sns.histplot(data=df, x="age", kde=True)
plt.figure(figsize=(20,10), dpi=200)

sns.histplot(data=df, x="age", hue="loan")

sns.hisplot(data=df, x="pdays", bin=40)
sns.histplot(data=df[df['pdays']!=999], x='pdays', bins=40)
df['contact'].unique()
df['duration']
```

M T W T F S DATE:

```
sns.histplot(data=df, x='duration', hue='contact')
plt.xlim(0,1000)

sns.countplot(data=df, x='contact')

sns.countplot(data=df, x='job', order=df['job'].value_counts().index)
plt.xtict(rotation=90):
plt.figure(" )
```

same for education

↳ check with hue='default'

```
df['default'].value_counts()

sns.pairplot(df)
```

Coding part #2

```
X = pd.get_dummies(df)
X.head()

st = StandardScaler()
= st.fit_transform(X)
scaled_X
```

MIDWIFES DATE:

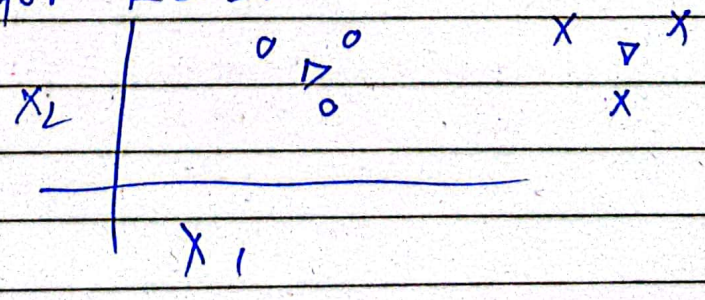
```
from sklearn.cluster import KMeans
help(KMeans)
↳ n_clusters = 8 default
max_iteration = 300 default

model = KMeans(n_clusters=2)
model.fit_predict(scaled_x)

clusters_labels = model.fit_predict(scaled x)
cluster_labels
  array([1,1,1,.... 0,0,0])

X['cluster'] = clusterlabels

X.corr()['cluster'].iloc[:-1].sort_values()
 .plot(kind='bar')
```

06 # **Coding part 3**

How to measure "goodness of fit"

- we could measure the sum of the distances from point to cluster centers.

for K = 2

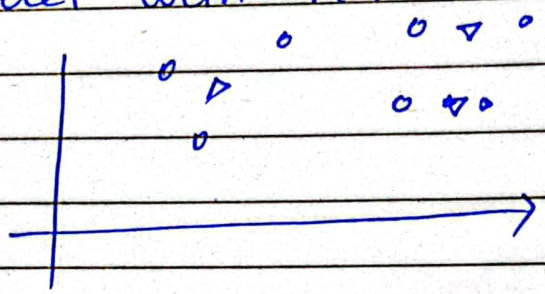


DATE:

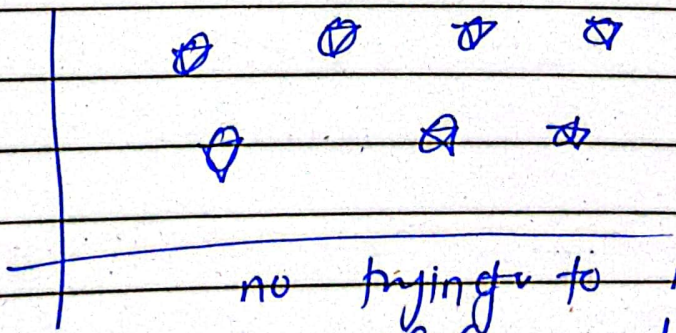
M T W T F S

- for all the points we measure the sum of squared distances from point to the cluster center:

- The we fit an entirely new k-means model with K+1:



- Then measure again the sum of the squared distance (SSD) to center.
- In theory this SSD would go to zero once K is equal to the number of points-
- You would have a cluster for each point! SSD would be perfect at 0!



no trying to minimize sum of squared distance but instead using as a measurement tool

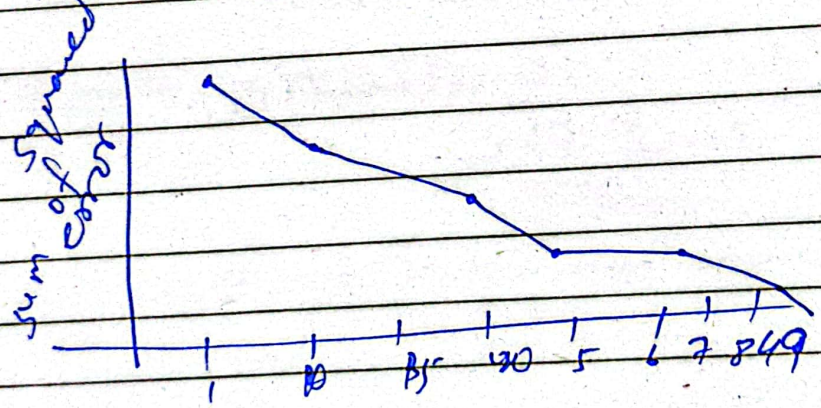
DATE:

MODULES

- We keep track of this SSD value for a range of reduction in SSD begins to decline.
- This signifies that adding an extra cluster is not obtaining enought clarity of clusters seperation to justify increasing K.

↓ This is known as the "elbow" method since we will track where decrease in SSD begins to flatten out compared to increasing K-values-

- Start with K=2
- Increase K and measure SSD



There points are strong indicator that increasing K further is no longer justified as its not revealing more "signal".

- You can also measure out this SSD in a barplot

#code section

M T W T F S DATE:_____

```
ssd = [ ]
for K in range(2,70):
    model = kmeans(n_clusters=K)
    model.fit(scaled_X)
    ssd.append(model.inertia)
```

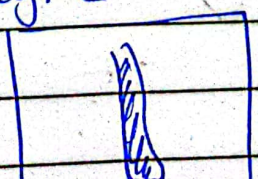
↓
#SSD points
cluster center

```
ssd
plt(range(2,10), ssd, 'o--')
pd.Series(ssd)
pd.Series(ssd).diff()
```

K-Means Clustering
Color Quantization
part One

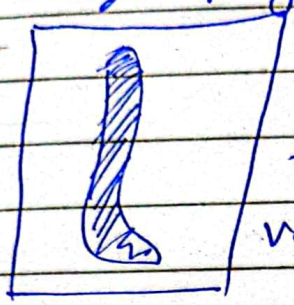
- One ~~image~~ interesting application of clustering is on image quantization.
- let discuss images, computers, colors, and quantization to get an idea of how K Mean clustering can be applied to different fields.

- Imagine an image of a single pen stroke:

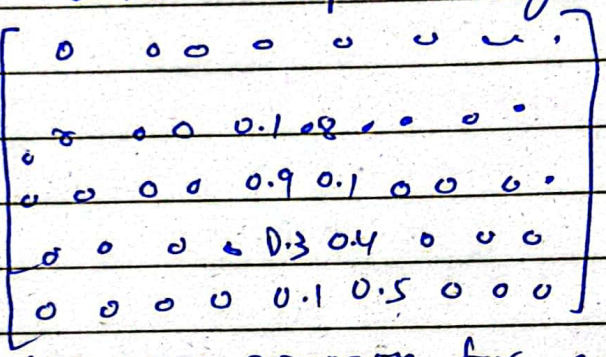


MIDTERMS DATE:_____

- This image is in grayscape, meaning the color ranges goes from black to white.



- A computer will store this information as an array with values between a range.
- Notice 0 is white and 1 is black, with values in between representing gray.



- It is also very common for computer to store value from 0-255 for scales.
- The range 0 to 255 has to do with how computer stores 8-bit numbers.

- Color Images can be represented as array, what about color images.
- Color images can be represented as a combination of Red, Green, Blue.

- Combining Green, Red, Blue to create colors

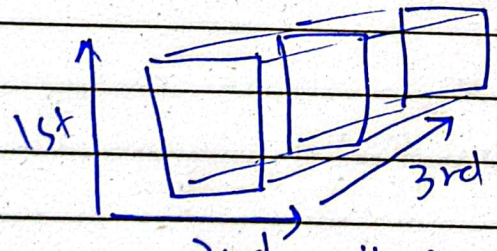
- From a computer perspective, this looks like 3 array, each array representing a color channel.

M T W T F S DATE:_____

- For example a single pixel (1 by 1 image) here is (213, 111, 56) for (R, G, B)

- The shape of the color arrays there has 3 dimensions -
  - Height
  - Width
  - Color channel

- If you read an image and check its shape - it will look like
  - (1280, 720, 3)
  - (1280 pixel width)
  - (720 pixel height)
  - (3 color channels)



- keep that in mind computer won't know a channel is Red, it just knows that there are now 3 intensity channel.

- From a computer perspective you simply have an array with 3 dimensions, where a user or display function can attribute each dimensions to a color channel (e.g. red intensity)

M T W T F S DATE:_____

- swapping these array accross channels would allow for effect such as color inversion.
- Now how can we apply clustering to RGB color channel and images?

Code part two

```
import numpy as np
import matplotlib.image as mpimg
image as array
mpimg.imread('... // path-tree.jpg')
plt.figure(dpi=200)
plt.imshow(image_as_array)
```

need numpy array

(H, W, C) ---- 2D

(H*W, C)

```
(h, w, c) = image_as_array.shape

image_as_array2d = image_as_array.reshape
                                (h*w, c)
Image.array

from sklearn.cluster import KMeans
model = KMeans(n_clusters=6)
label = model.fit_predict(image_as_array2d)

labels
```

M T W T F S DATE:

model.cluster_centers_

rgb_codes = model.cluster_centers_.round(0).astype(int)

rgb_codes[labels]

= np.reshape(rgb_codes[labels], (h, w, c))

quantized image

plt.imshow(quantized_image)