Almost every major tech company has applied recommendation system them in some form. Amazon uses it to suggest products to customers, YouTube uses it to decide which video to play next on autoplay, and Facebook uses it to recommend pages to like and people to follow.

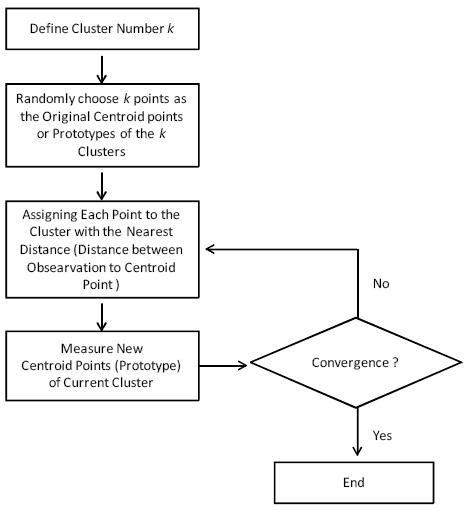
What's more, for some companies like Netflix, Amazon Prime, Hulu, and Hotstar, the business model and its success revolves around the potency of their recommendations.

***Netflix even offered a million dollars in 2009 to anyone who could improve its system by 10%.***

There are also popular recommender systems for domains like restaurants, movies, and online dating. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services. YouTube uses the recommendation system at a large scale to suggest you videos based on your history. For example, if you watch a lot of educational videos, it would suggest those types of videos.

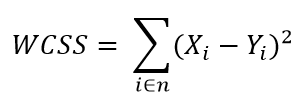
Medication errors are one of the most serious medical errors that could threaten patients’ life. More than 42% of these errors are caused by doctors who have limited experiences/knowledge about drugs and diseases. Another reason lies in the increasing number of available drug information, which has brought obstacles concerning the discovery of relevant drugs and drug-disease interactions. In this context, drug recommender systems have been developed to assist end-users and healthcare professionals in identifying accurate medications for a specific disease.

**k-means clustering**



**#Elbow Criterion Method:**

Elbow method is one of the robust one used to find out the optimal number of clusters. In this method, the sum of distances of observations from their cluster centroids, called **Within-Cluster-Sum-of-Squares (WCSS)**. This is computed as



*Yi* is centroid for observation *Xi*.

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

X = load\_iris().data

y = load\_iris().target

wcss=[]

for i in range(1,10):

kmeans = KMeans(i)

kmeans.fit(X)

wcss\_iter = kmeans.inertia\_

wcss.append(wcss\_iter)

number\_clusters = range(1,10)

plt.plot(number\_clusters,wcss)

plt.title(' The Elbow Method showing the optimal k')

plt.xlabel('Number of clusters')

plt.ylabel('WCSS')

**#Silhouette Coefficient score**

If the ground truth labels are not known, evaluation must be performed using the model itself. The Silhouette Coefficient (**[sklearn.metrics.silhouette\_score](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html" \l "sklearn.metrics.silhouette_score" \o "sklearn.metrics.silhouette_score)**) is an example of such an evaluation, where a higher Silhouette Coefficient score relates to a model with better defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores:

* **a**: The mean distance between a sample and all other points in the same class.
* **b**: The mean distance between a sample and all other points in the next nearest cluster.

The Silhouette Coefficient s for a single sample is then given as:

s=b−a/max(a,b)

The Silhouette Coefficient for a set of samples is given as the mean of the Silhouette Coefficient for each sample.

**/\*A higher Silhouette Coefficient score relates to a model with better-defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores: `**

**a: The mean distance between a sample and all other points in the same class.**

**b: The mean distance between a sample and all other points in the next nearest cluster.**

**The Silhouette Coefficient is for a single sample is then given as:**

**s=b-a/max(a,b)\*/**

from sklearn.metrics import silhouette\_score

from sklearn.datasets import load\_iris

from sklearn.cluster import KMeans

X = load\_iris().data

y = load\_iris().target

for n\_cluster in range(1, 10):

kmeans = KMeans(n\_clusters=n\_cluster).fit(X)

label = kmeans.labels\_

sil\_coeff = silhouette\_score(X, label, metric='euclidean')

print("For n\_clusters={}, The Silhouette Coefficient is {}".format(n\_cluster, sil\_coeff))

"""

**K-Means Clustering: Traffic Analysis**

"""

import pandas as pd

import pylab as pl

from sklearn.cluster import KMeans

from sklearn.decomposition import PCA

variables = pd.read\_csv(' /content/London.csv')

Y = variables[['CarsTaxis']]

X = variables[['PedalCycles']]

X\_norm = (X - X.mean()) / (X.max() - X.min())

Y\_norm = (Y - Y.mean()) / (Y.max() - Y.min())

pl.scatter(Y\_norm,X\_norm)

pl.show()

#Elbow curve

Nc = range(1, 20)

kmeans = [KMeans(n\_clusters=i) for i in Nc]

kmeans

score = [kmeans[i].fit(Y\_norm).score(Y\_norm) for i in range(len(kmeans))]

score

pl.plot(Nc,score)

pl.xlabel('Number of Clusters')

pl.ylabel('Score')

pl.title('Elbow Curve')

pl.show()

#Principal Component Analysis and K-Means

pca = PCA(n\_components=1).fit(Y\_norm)

pca\_d = pca.transform(Y\_norm)

pca\_c = pca.transform(X\_norm)

kmeans=KMeans(n\_clusters=6)

kmeansoutput=kmeans.fit(Y\_norm)

kmeansoutput

pl.figure('6 Cluster K-Means')

pl.scatter(pca\_d[:, 0], pca\_c[:, 0], c=kmeansoutput.labels\_)

labels=kmeansoutput.labels\_

labels

pl.xlabel('Cycles')

pl.ylabel('Cars and Taxis')

pl.title('6 Cluster K-Means')

pl.show()