FACE SIMILARITY PERCENTAGE

# Overview

This project’s aim is to create the backend for an application to detect how similar different faces are.

# Methodology

1. Gather Data
2. Find faces and crop them out and use only faces
3. Covert faces to face encodings[T&E HERE: ENCODING SIZE]
4. Cluster images[T&E HERE: CLUSTERING ALGO]
5. For similarity score: calculate distance and get score(1/(1+distance)) [T&E HERE: DISTANCE FORMULA]
6. Evaluate model

{PLUS ULTRA ZONE}

1. Create web interface where users can upload images and find similarity between the images

# Plan Details

## Gather data

Google photos download

## Find faces

Methods to try out for face detection:

* Face detection api
* Haar cascades
* Pre-built CNN

Save all ROIs from images

## Convert faces to face encodings

* Face\_recognition.face\_encodings
* NN

## Cluster Images

Clustering using :

* chinese whisper
* DBSCAN
* Kmeans
* Affinity Propagation
* BIRCH
* Mean shift
* OPTICS
* Spectral clustering
* HAC

## Similarity score

For similarity score: calculate distance and get score(1/(1+distance

Calculating distance in:

* Euclidian Distance
* Manhattan Distance
* Chebychev distance
* Minkowski Distance
* Canberra Distance
* Hamming Distance
* Mahalanobis Distance
* Pearson Correlation
  + Absolute Pearson
  + Un-centred Correlation
  + Absolute Un-centred
* Eisen-Cosine
* Spearman and Kendall

## Evaluating Model

Evaluating the model and see how good the quality is:

* Rand Index
* Adjusted Rand index
* Fowlkes Malows Score
* Silhouette score
* Calinski Harabaz Index

## Web App

Use Django to create a web app/netlify to deploy

### Features

* User can upload two images
* Preview of images
* Show faces on images
* Write similarity score beneath image
* Reset option
* Label images from user end
* For pictures with multiple faces, have user select which faces to see similarity between