Project Description: **Beauty Detector.** This project will utilize various machine learning algorithms to model the society’s general idea about beauty. It will use several online datasets that collect people’s opinion about the most beautiful person they’ve seen. The finished project will be able to recognize people’s face and score the level of beauty of the given picture.

Competitive Analysis: I found several online beauty detectors, and some of them ask me to find the critical points, including the position of eyes, nose, and elbows manually. But for my project, I plan to train the computer to find the critical points by themselves. One similarity of these apps and my project will be that the result of the detector varies greatly when using different datasets. Because beauty is a really subjective topic, so the actual importance and potential of this project is not to find the most beautiful human being in the world, but to utilize specific algorithms to locate the critical points of a given human face, and to follow the standard set by the given dataset.

Structural Plan: The finalized project will contain two large part, the machine learning part and the open CV part. The open CV part will be relatively small and short because it will only be responsible for capturing picture by camera and save the picture or upload the picture to go through the machine learning algorithm (one file). Facial recognition and feature extraction will be performed by Haar Cascades (one file). But it turns out that the Haar Cascade method is not accurate enough, so I plan to use convolution instead. The machine learning part will be the focus of this project. It will include classification algorithms, such as Random Forest, Support Vector Machines, K-Nearest Neighbors, and Logistic Regression, to build up the beauty classifier (2-3 file).

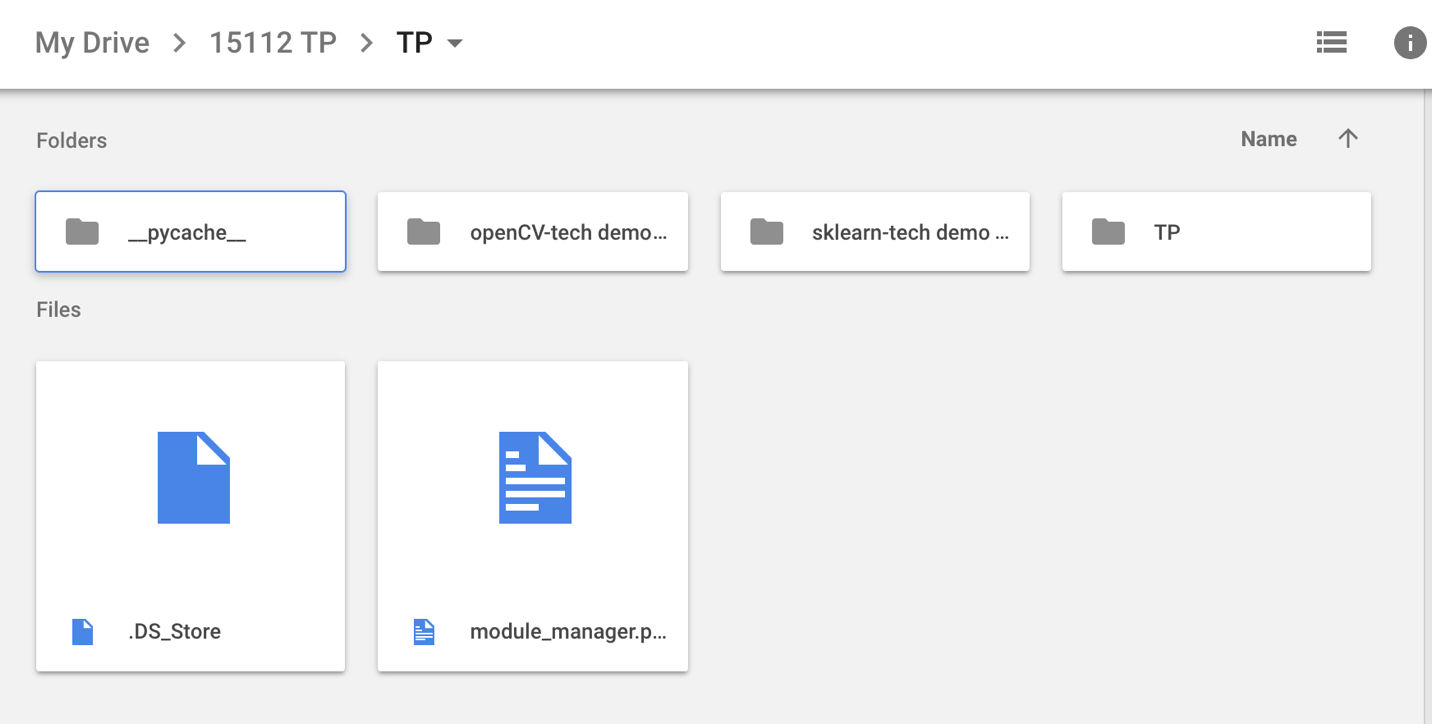
Algorithmic Plan: The trickiest part in this project will be the beauty classifier part. I plan to use classification because it’s suitable for many input variables. After the facial features were extracted by Haar Cascade/convolution, a feature matrix will be created as the input for the classifier. The feature matrix will include around 10 features that related most to the human ratings of beauty. After generating the matrix, I plan to use the random forest algorithm instead of others listed above. The reason is that compared to SVM, RF treats features unequally by discarding irrelevant attributes and using discriminative features more frequently, which is suitable for this project because there’re some facial features that people value more than others. Algorithms like SVM will treat all features equally and therefore is very sensitive to high dimensional data, which will result in large fluctuation when there’s no necessity. Regression algorithms are ruled out because they spend all efforts in finding general relationships between features and labels, but they can’t find interactions between multiple variables directly. I may try to write up a KNN algorithm depending on the actual dataset I found because KNN’s performance varies largely when the given dataset changes.

Timeline Plan:

TP1: finish the facial recognition and facial feature extraction part, determine the dataset, start to approach the RF/KNN algorithm,

TP2: finish the classifier, start to use OpenCV to capture picture and think how to connect the camera to the scoring system of the classifier

TP3: be able to use the camera to capture a picture, upload the picture to the classifier and get the score shows up on the picture

Version Control Plan: I plan to use google drive to store all my files and codes (see screenshot below). 

Module List: sklearn, OpenCV, numpy, pandas, csv, matplotlib, seaborn

TP2 Update:

The same as before.

TP3 Update:

The final version of this version uses tkinter to combine the machine learning and OpenCV part. I use a total of 8 machine learning models: decision tree regressor, decision tree regressor with Ada Boost, linear regressor, linear support vector regressors, support vector regressors with linear kernel, support vector regressors with polynomial kernel, support vector regressors with rbf kernel, and random forest regressors. The database used to train these models are “SCUT-FBP5500\_with\_Landmarks” from South China University of Technology, which includes 5500 people’s faces from around the world, their facial landmarks, and their face score from around 5 to 10 people. There’re 86 key points per person, but I only make use of 20 of them because the Haar Cascade method cannot locate other minor points. Also, the feature matrix of my model includes 17 columns of data, which measures the ratio of the width of face to the distance of the pupils, the ratio of the height of the face to the distance of mouth and nose, etc. Since the width and height of face are the largest data, the ratio is less than 1, so for each data in the feature matrix, I use the value of itself times 100 to make the regressors work as expected.

There’re also some new modules added in the final version. Here is the complete module list: numpy, matplotlib, pandas, seaborn, sklearn, pickle, cv2, tkinter, pillow(PIL), string, random, pyscreenshot, os, re, struct, shutil, csv, math.