Springboard – Data Science Camp
Capstone Project 2

Fútbol Match Highlights

By Tom Widdows
April 21, 2020

Table of Contents

(1) Introduction

<u>Problem</u>

College coaches require soccer players to submit highlight video and game film to be considered for an athletic scholarship. The issues facing student-athletes to provide highlight videos and game films include recording games is time-consuming, athletes must rely on someone else to take the video, most people would prefer to watch the game and not be a videographer, processing the game file is time consuming and the required hardware and software are expensive.

<u>Stakeholders</u>

Stakeholders include high school athletes, their parents and college sports recruiters.

<u>Results</u>

The goal is to identify a target player in a photo (ultimately a video) with a high degree of accuracy. By managing the quality throughout the process, the resulting accuracy of locating (and ultimately tracking) the target player should be very high.

2019
OSAA
OREGON SCHOOL ACTIVITIES ASSOCIATION
3A/2A/1A GIRLS SOCCER CHAMPION
OnPoint

Figure 1 – Athletes and Parents

<u>Post project</u>

Two videos will be stitched together, face images will be extracted from the videos (which are images), the face recognition utilized in this project will be applied to the video image and the target player identified allowing the software to effectively track the player.

<u>Implementation Details</u>

Additional implementation details and all code related to the project can be found on my GitHub repository:

https://github.com/8-Waste/Springboard/tree/master/Capstone%202%20-%20F%C3%BAtbol

(2) Approach

<u>(2.1) Data Acquisition and Wrangling</u>

Facial recognition requires a sequence of related steps including ensuring the source image is sharp, locating faces in the image, provide quality checks to ensure quality faces, centering and measuring unique facial features, comparing the target face to known faces and making a prediction.

The data source for this project is images of fútbol players taken by Tom Widdows. Although the images are from still photographs, the application should be easily enhanced to work with frames of a video. The images are primarily in a proprietary Canon format. After consolidating the photos, they are converted, if necessary, from the proprietary Canon format (.CR2) into JPG files so we can easily access them in python.

To evaluate image sharpness, A support vector machine (SVM) that uses labeled photos (sharp and blurred) with 12 features consisting of 3 measurements (mean, variance and maximum) each of 4 edge detection algorithms (laplace, sobel, roberts and canny) is fit and saved. The source images, with the same 12 calculated features, are processed through the trained model and identified as a sharp or blurry image.

(2.2) Storytelling and Inferential Statistics

The sharp images are processed through two separate face detection algorithms, a Harr-cascade classifier (HARR) included with OpenCV and a Multi-task Cascaded Convolutional Network (MTCNN).

The images below (Figure 2) were created using the HARR classifier for face detection. The sample images below certainly show the HARR classifier identified many non-faces as faces (false positives).

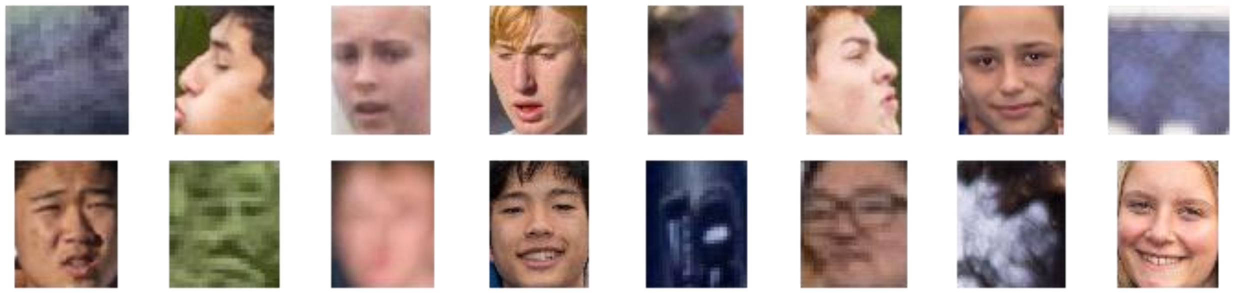

Figure 2 – Faces from HARR Classifier

The images below were created using a Multi-task Cascaded Convolutional Network (MTCNN) for face detection. The MTCNN identified many non-faces as faces (false positives) but at first look, appears to have performed better than the HARR classifier.

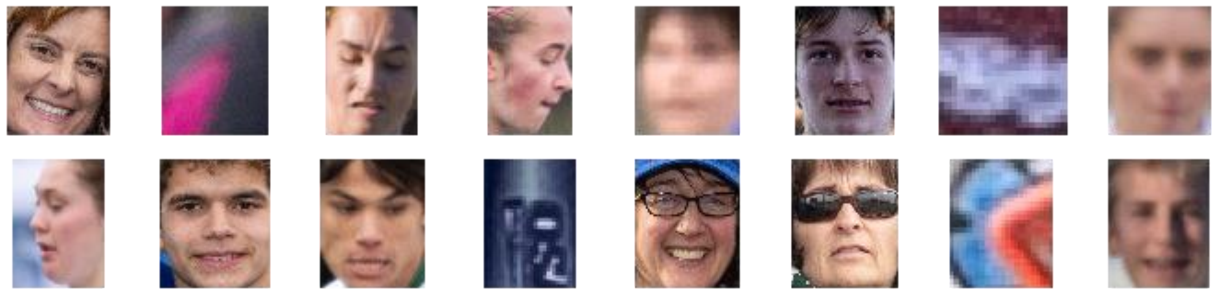

Figure 3 – Faces from MTCNN Classifier

A face in an image is rarely more than 1% of the entire image (calculated using square pixels) and the median percentage is 0.20%.

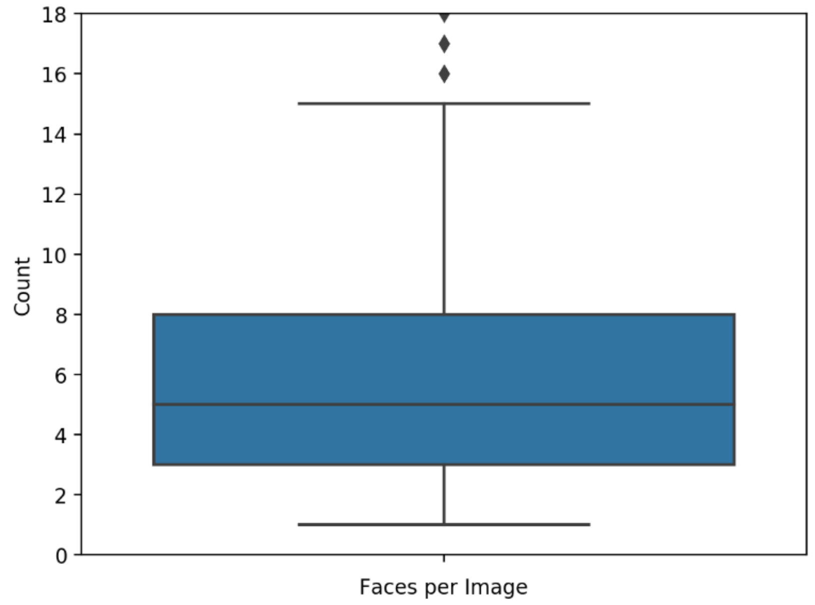

Figure 4 – Faces per Image

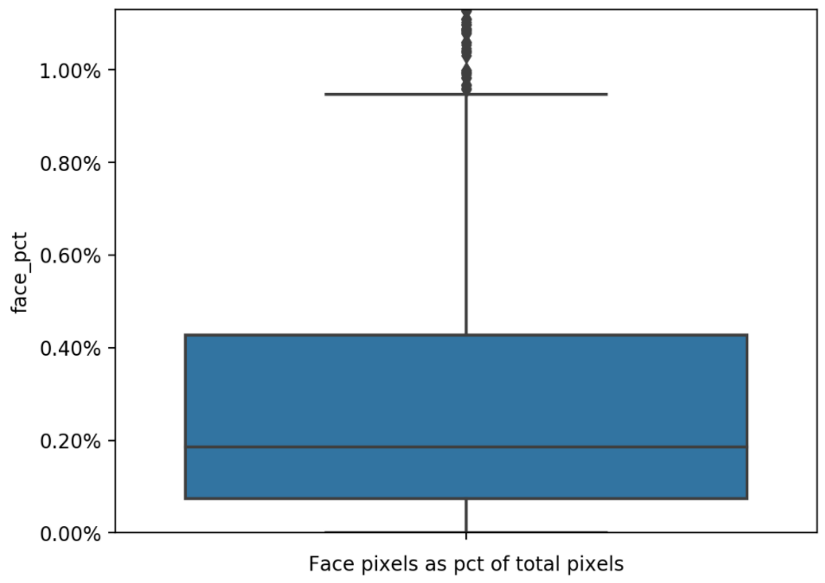

Figure 5 – Face Percent of Total Pixels

An image rarely contains more than 15 faces and the median number of faces in an image is five.

Now let's examine some of the color properties of an identified face. We will calculate the average color of the face, the top 5 dominate colors in the face and the percent of skin tones in the face. Below are 3 random faces with the calculated color values:

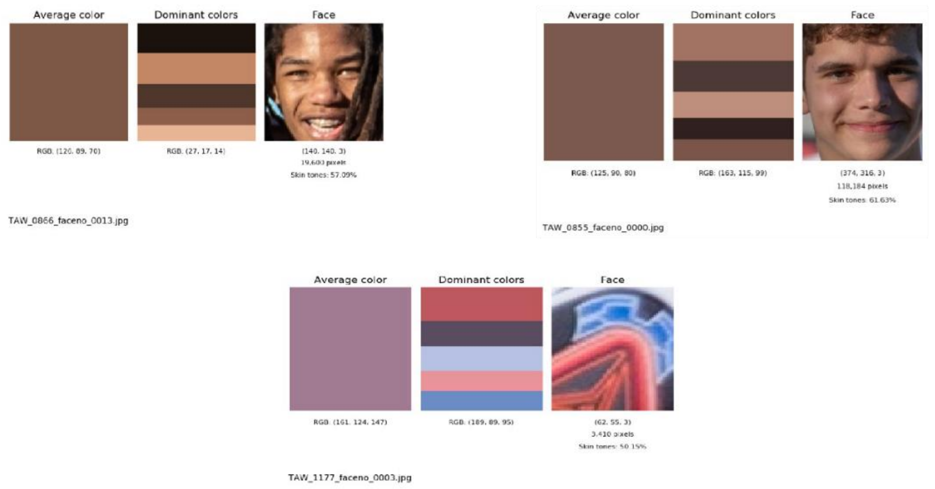

Figure 6 – Image Color Distribution

.

Let's calculate if the difference in skin tones between face and non-face images is statistically significant. In each image, we will programmatically count the number of faces and take an equal number of non-faces (or more accurately a random area the same size as the face). Since the median face covers only 0.20% of an image and rarely covers over 1% of an image, the result should include very few faces and it would be unlikely the face would be centered. Below is a sample of the calculated non-faces:

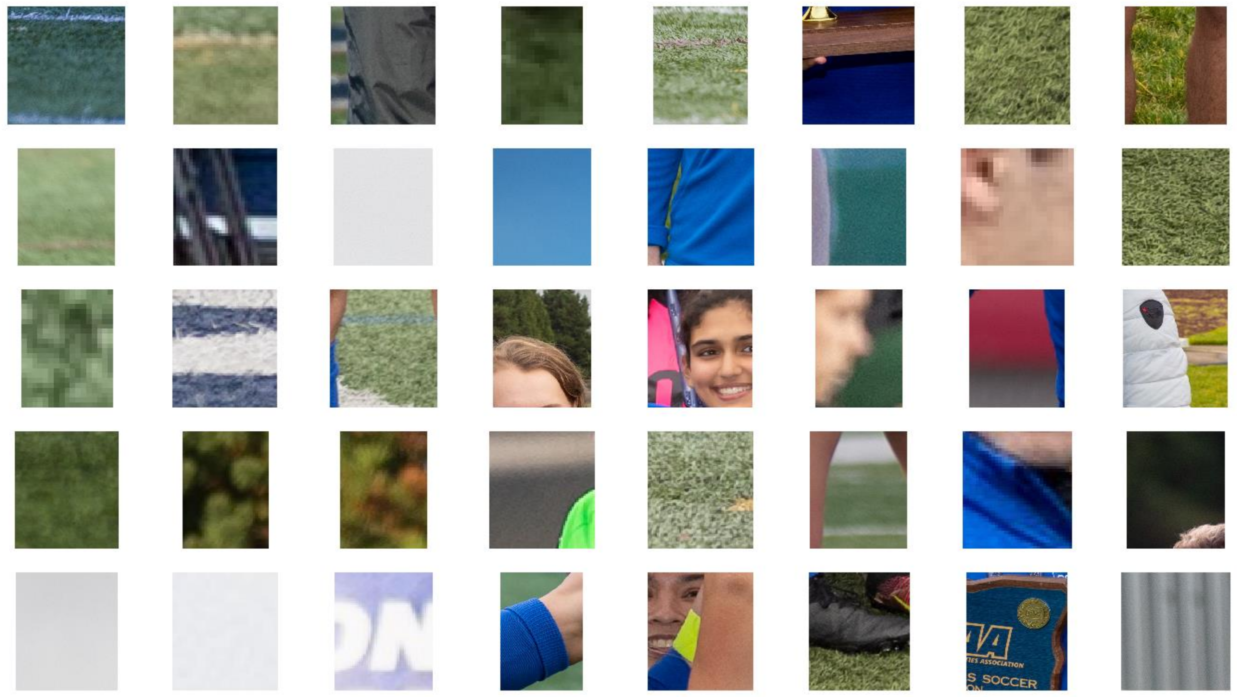

Figure 7 – Object (non-face) images

Now let’s examine some of the color properties of a non-face. Again, we will calculate the average color of the non-face, the top 5 dominate colors in the non-face and the percent of skin tones in the non-face. Below are 3 random non-faces with the calculated color values:

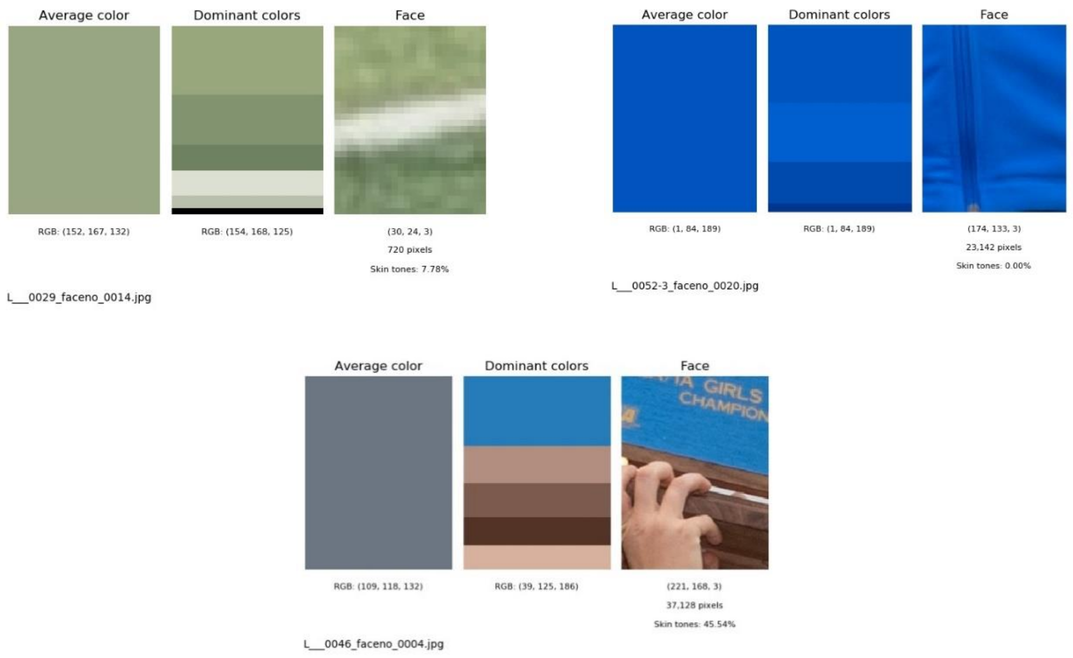

Figure 8 – Color Properties of Non-Face Images

Let’s test if the means of the percent of skin tones are identical. The hypothesis test is to test if the difference in percent skin tones in a face image is statistically significant. The null and alternative hypothesis are as follows:

H_0: Percent Skin Tones has no effect on face identification, means are same

H_1: Percent Skin Tones has effect on face identification, means are different

Samples:	**931**
Mean:	**82.39% (dashed red line)**
Std Dev:	**16.86%**
Median:	**87.69%**

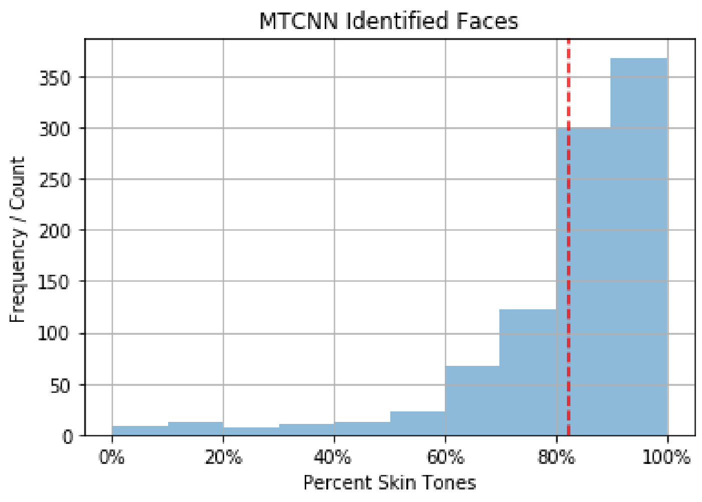

Figure 9 – Percent Skin Tones

Samples:	**931**
Mean:	**19.72% (dashed red line)**
Std Dev:	**26.71%**
Median:	**5.98%**

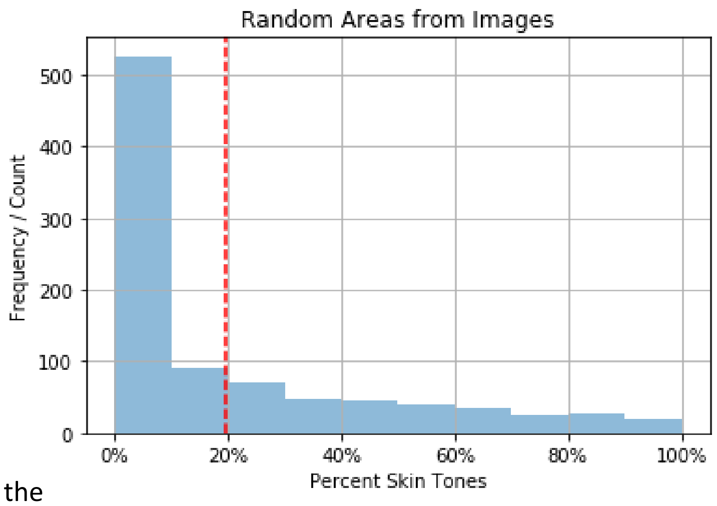

Figure 10 – Percent Skin Tones

I calculated a two-sided t-test for the null hypothesis that the samples have identical average (expected) values utilizing a significance level of .01. The p-value was 0.0000000 so **I rejected the null hypothesis in favor of the alternative hypothesis.**

H_1: Percent Skin Tones has effect on face identification, means are different

(2.3) Baseline Modeling

The application takes a source image and processes it through the **Face Detector** model to identify the location of the faces (red squares). The application processes the pixels inside the red boxes through the **Face vs Object Model** to remove non-faces. The surviving faces are then processed through the **Focused vs Unfocused Model** to eliminate out-of-focus images. The surviving faces are then compared to the target face which is done by the **Recognizer Model**. The Recognizer Model utilizes a pretrained deep neural network (DNN) from OpenFace implemented in Python and Torch for face recognition.

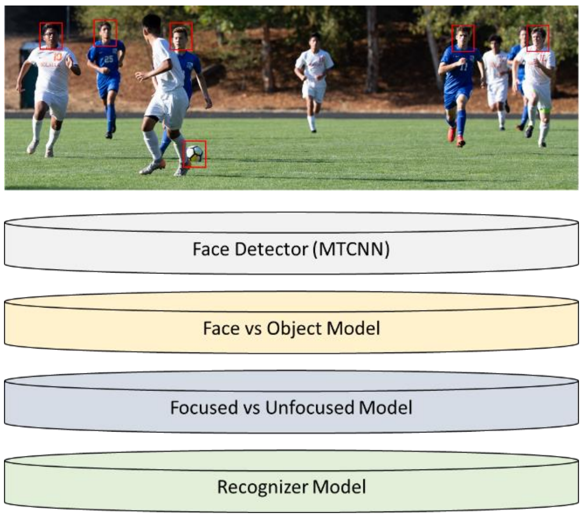

Figure 11

There are 5 steps (notebooks) to successfully run this application. Steps 1, 2 and 3 involve training and saving models that are used in steps 4 and 5 (Figure 12).

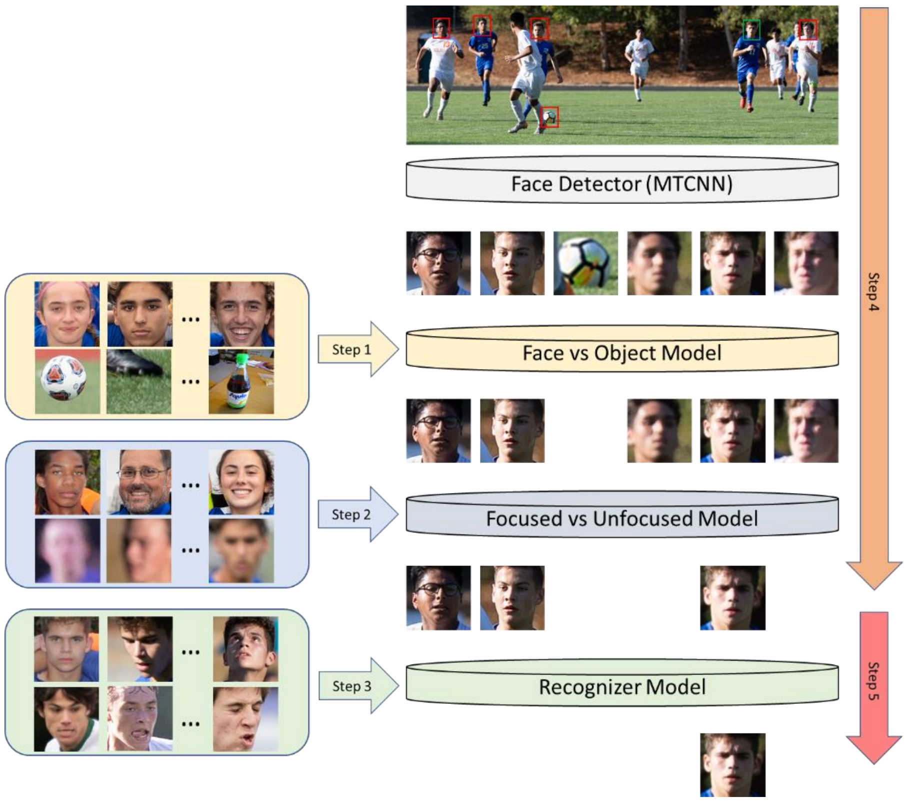

Figure 12

(2.3.1) Face vs Object (non-face)

The Harr-cascade classifier (HARR) included with OpenCV and a Multi-task Cascaded Convolutional Network (MTCNN) classifier were excellent at identifying in and out of focus faces, but the classifiers incorrectly identified many objects as faces (Figure 13). As discussed in Storytelling and Inferential Statistics, skin tone was determined to be statistically significant in identifying faces vs non-faces. I found that training a model on Faces vs Objects (non-faces) was more effective then evaluating on skin tone.

I created a dataset of 200 faces and 264 objects (non-faces). The objects consisted of 264 random patches from my existing photos. I trained using a KNeighborsClassifier classifier and obtained a True Positive Rate (TPR) / Recall rate of 88.89% and 85.00% on my validation data and test (holdout) data respectively.

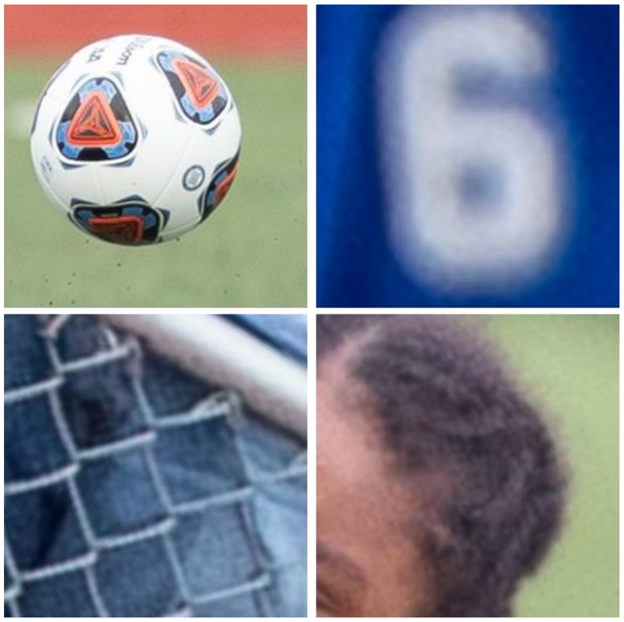
Figure 13

(2.3.2) In-Focus vs Out-of-Focus Images

As discussed in Data Acquisition and Wrangling, I initially trained a Support Vector Machine (SVM) that used labeled photos (sharp and blurred) with 12 features consisting of 3 measurements (mean, variance and maximum) where each have a 4 edge detection algorithm (laplace, sobel, roberts and canny). My source photographs, with the same 12 calculated features, were processed through the trained model and labeled as in-focus or out-of-focus. The issue discovered with this approach is quality photographs are taken with a low aperture that makes the in-focus area very sharp and the rest of the photo intentionally more out-of-focus. In Figure 14, player #11 is in very sharp focus but the referee's face is completely out-of-focus and unrecognizable. The SVM classified this photo as out-of-focus which technically, for the photo as a whole, could be considered correct but eliminates a very sharp, in-focus face.

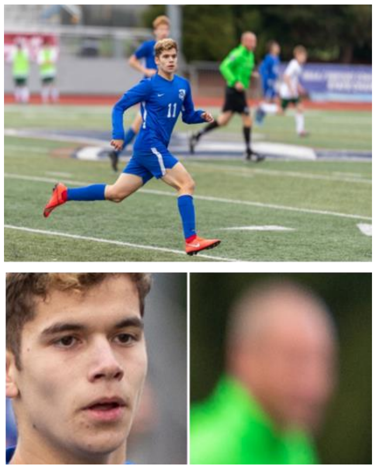
Figure 14

My revised apporach checks the focus of faces extracted from the images. I created a dataset of 1,701 out-of-focus faces and 655 in-focus faces. I trained using a GaussianNB classifier and obtained a True Positive Rate (TPR) / Recall rate of 85.99%, and 83.33% on my validation data and test (holdout) data respectively.

(2.3.3) Target Face Embedder

My dataset consists of 114 images of the target player and 566 images of other people. All the images are RGB (256, 256, 3). For preprocessing, I aligned all the faces in the images and utilized blobFromImage (mean subtraction and scaling) to prepare the photos for creation of an 128d vector face identifier utilizing a pretrained model based on OpenFace. The model was trained utilizing LBGM classifier utilizing a training, validation, and test set. The model was optimized utilizing Hyperopt using Kappa as a metric to maximize.

2.4 Extended Modeling (Pending)

(2.4.1) Face vs Object (non-face)

(2.4.2) In-Focus vs Out-of-Focus Images

(2.4.3) Target Face Embedder

(3) Findings (Pending)

(3.1) Face vs Object (non-face)

(3.2) In-Focus vs Out-of-Focus Images

(3.3) Target Face Embedder

(4) Conclusions and Future Work (Pending)

(5) Client Recommendations (Pending)

(6) Consulted Resources (Pending)