Is Beauty Really in the Eye of the Beholder?

Yun (Albee) Ling (yling), Jocelyn Neff (jfneff), and Jessica Torres (jntorres)

Abstract

Recent research suggests that high facial symmetry plays a large role in whether or not a person is deemed beautiful. To test this hypothesis, a beauty classifier was built utilizing various machine learning algorithms in hopes of modeling society's general idea of beauty. Data collection included survey responses and online facial images. Facial recognition and facial feature extraction was performed with Haar Cascades. Beauty classifiers consisted of algorithms such as Random Forest, Support Vector Machines, K-Nearest Neighbors, and Logistic Regression. Facial features were detected with 0.02 error, and beauty was correctly classified using the Random Forest model with 0.21 error.

1 Introduction

A common belief is that more symmetric faces are more beautiful. If this is true, could it be possible that beauty is not subjective, but in fact a result of a mathematical compilation of facial features? By applying machine learning to face detection, facial feature extraction, and the quantification of facial symmetry, a model theoretically could be built that attempts to predict a face's beauty based off features alone. Using features that represent the symmetry of a face, a model's high accuracy could suggest that society's idea of beauty can be learned by a machine. Moreover, a model's poor performance could highlight intrinsic qualities of beauty that cannot be quantified in model features alone.

2 Materials and Methods

2.1 Data

2.1.1 Kaggle

Kaggle provided 7,048 96x96 gray-scale images with the pupils, eye corners, eyebrow corners, and mouth corners identified. Each image was a head shot of a face. The images were not necessarily head-on shots, and the subjects had varied facial expressions. In addition, there was only a total of 50 subjects: each subject had multiple photos of him or her with a variety of poses and facial expressions. In order to train the Haar Cascades Classifier, the Kaggle dataset was partitioned into 70% training and 30% testing.

2.1.2 Survey

Fourteen celebrity images were selected to be distributed in a survey asking users for their assessment of the celebrity's beauty. Widely held opinions of a celebrity's attractiveness could impact responses, so lesser-known celebrities were chosen. In addition, the set of celebrities was limited to Caucasians to reduce biases caused by ethnicity. There were equal number of males and females. Each image was selected so that each facial image was forward facing with little facial tilt. These qualifications eased facial detection, ensuring that the features and distances derived from these facial features were correct. Each celebrity was ranked on a scale of 1 to 10. With 242 responses, there were a total of 3,388 rankings of facial beauty. The mean rating across all faces was 6.22. The rating distribution for each face followed a Normal Distribution where the mean varied from 4.26 to 8.23. These photos served as the test set for the models later developed (Figure 1).

2.1.3 Online Images

In addition to the celebrity images, 100 images were collected of faces online to serve as the training set for the models. These images also served as the test set for the Haar Cascade feature detector: in order to train the models, facial features first needed to be identified. For each of these images, a binary classification of beautiful or not beautiful was given. These images, like the celebrity images, were cropped to 96x96 pixels so that there would be more resemblance between the Kaggle training set and the test set.

2.2 Facial Feature Extraction

Using the Haar Cascades facial feature detector provided by OpenCV [8], separate models were developed to detect the bounding box for the face, eyes, nose, and mouth. OpenCV had a face and eye detector, but lacked a mouth and nose detector. In order to detect the nose, a Haar Cascade model was trained. It was trained on 45 positive images and 100 negative images selected from the 70% of the Kaggle data labeled as training data. For the developed model, a window size of 24 x 24 pixels was used with 5 stages of training, a minimum hit rate of 0.99, and a max false alarm rate of 0.5. The positive images were tight bounding boxes of the facial features to be detected.

In a eyeball comparison of the developed and available model, the available model had a tighter and more accurate bounding box for the features. Therefore, the online model was used for the test set feature extraction. Similar findings were found for a nose detector online [4].

In order to minimize false positives, the search space for the classifier at each stage was minimized. First, the bounding box for the face was identified. Then, the pixel range for searching for the eyes, nose, and mouth was reduced to within the box. The eye model resulted in the largest number of false positives, so additional constraints were implemented. Given that all faces were 96x96 pixels and the faces were constrained to all be similar facial shots, the search space was reduced to the upper 30 pixels of the face. If the detector still found more than two "eyes", all boxes were considered questionable and the classifier did not classify the eyes. If, however, the feature detector found one or two "eyes", then the eye with the lower x-axis pixel location, i.e. the top left corner of the box was further to the left side of the picture, then that box was deemed to be the right eye (left and right side of the face are from the subject's point of view).

Nose and mouth were also detected. The mouth classifier was likely to return many "mouths", so only the bounding box that had the largest y coordinate (and thus furthest to the bottom of the picture) was classified as the mouth.

2.3 Feature Matrix Creation

After facial features were detected, a feature matrix was created as input for beauty classification models. In order to

calculate various distance statistics, coordinates for the center of boxes that enclose each of the facial features (eyes, nose and mouth) were identified. The final feature matrix included basic facial features that are commonly cited in literature [6], such as length and width of the nose. In addition, attributes that are concerned about symmetry were added, since our hypothesis is that facial symmetry is associated with beauty. Specially, two measurements that were cited in multiple literature [5,7], Central Facial Asymmetry (CFA) and Facial Asymmetry (FA) were part of our feature matrix. FA is the sum of pairwise differences between midpoints of each facial feature in the x-axis direction, while CFA is the sum of differences between midpoints of adjacent facial features. Figure 2 is a visualization of all the distance statistics calculated for each face.

2.4 Classification

Four classification algorithms implemented in R: Logistic Regression (LR) [1]; Support Vector Machine (SVM) [1], K-Nearest Neighbor and, Random Forest (RF) [1]. All classification models were trained on Leave One Out Cross Validation (LOOCV) method.

2.4.1 Feature Selection

The original feature matrix included 40 features. A reduced feature matrix included 10 features were the most correlated with human ratings of beauty in the 100 images using "FSelector" Package in R. [1] All machine learning models were implemented in both feature matrices for comparison. P values of FA and CFA for each of the 14 celebrity faces were output based on the 100 training images. The p-values were calculated by dividing the number of training faces with the same or lower values of FA or CFA as the test example by 100.

2.4.2 Random Forest

Random Forest is a classification method utilizing combinations of tree predictors such that each tree is built independently from the others. A single classification tree is often not an accurate classification function. Thus, RF utilizes an approach which aggregates several inefficient classification trees using a "bagging'" procedure: at each step of the algorithm, several observations and several variables are randomly chosen and a classification tree is built from this new data set. For example, given a training set $X = x_1, \ldots, xn$ with responses $Y = y_1, \ldots, yn$, bagging repeatedly selects a random sample with replacement of the training set and fits trees to these samples. The result is a final classification decision obtained by either a majority vote on all the classification trees or by averaging the predictions from all the individual regression trees for unseen samples, x': $\hat{f} = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(x')$, where $B = b_1, \ldots, bn$ sampling. For our purposes here, majority vote on all classification trees was taken. The package "rpart" from R was used in generating our random forest model.

2.4.3 Logistic Regression

Logistic regression (LG) is a classification parametric model which is used to estimate the probability of specific observations to belonging to a particular class. The built in function "glm" from R was used to build our binary LG model assuming a binomial distribution. LOOCV was performed with the package "boot" package in R as well.

2.4.4 Support Vector Machines (SVM)

SVM is a supervised learning technique that finds the maximum margin separating hyperplane between classes. Kernel tricks are often applied to map input features into high dimensional space, especially when data are not linearly separable. Function "ksvm" in R package "kernlab" was utilized in all analyses. [3] Various SVM models and kernels were applied to the entire feature matrix with 40 features per face and the reduced feature matrix with only 10 most correlated features. The SVM models include C-SVM classification (C-svc), nu-SVM classification (nu-svc), and bound constraint-SVM classification (C-bsvc). The kernels include radial Basis kernel function "Gaussian" (rbfdot), polynomial kernel function (polydot), linear kernel function (vanilladot), hyperbolic tangent kernel function (tanhdot), Laplacian kernel function (laplacedot), and ANOVA RBF kernel function (anovadot).

2.4.5 K-Nearest Neighbor

KNN is an unsupervised learning algorithm that takes input integer k. For a given test example, the classification comes from the majority vote of the k data points that are the closest in Euclidean distance. Ties are broken at random. This method was chosen because its excellent performance in a previous study. [6] Function "knn" in R package "class" was used in all analyses using all possible values of k. [1]

3 Results

3.1 Facial Feature Extraction

The combination of feature detectors collected yielded a successful facial feature detector model. Table 1 summarizes the results from the Haar cascade classifiers developed and used. The training error was the error from testing on Kaggle 1,400 images where faces were not necessarily forward-facing, untitled, relaxed expressions. To determine if the algorithm correctly found the feature, the x,y coordinate specified by Kaggle for the feature had to lie within the bounding box the feature detector found. There were two test sets: the 100 images that would then comprise the training set for the beauty models, and the 16 celebrity images that comprised the test set for the models. Because neither of these sets of images had feature locations identified, a correct classification was eyeballed by running through the results. Accuracy was much higher, as all the testing images were forward facing, minimal tilt, relaxed faces. In the case that the model did not find a feature, no prediction was made. Given that there were only 116 test images, predictions that were not made by the feature detector were manually tagged in order to provide all features to the beauty models.

3.2 Feature Selection

The 10 most correlated features are distance between right pupil and top of forehead, distance between midpoint of two pupils and top of forehead, distance between midpoint of the forehead and midpoint of two pupils, distance between nose tip and left corner of mouth, distance between right pupil and right edge of forehead, distance between left pupil and left edge of face, FA, distance between midpoint of two pupils and nose tip, CFA, and distance between two pupils/Length of face. CA and CFA, which measure the overall asymmetry

of each face, are among the most associated features. Furthermore, the average values for CA and CFA are both smaller in people who were rated beautiful. This is evident that the most beautiful people have more symmetric faces. Among the seven celebrities that were deemed beautiful, 3 have significantly symmetric faces. However only 1 out of the seven celebrities that were not rated beautiful has symmetric face based on the p-values.

3.3 KNN

KNN was first implemented using all 40 features in the feature matrix. It achieved the lowest training error of 0.48 with k=42 of 46. Eight and nine out of the 14 final test examples were predicted successfully. Next, KNN was applied using only 10 features that were the most correlated with human ratings of the 100 training images. This classifier performs the best when k=3 or 5 and the training error 0.36. Using k = 3 and 5, KNN was able to accurately classify ten and nine of the 14 celebrity faces accurately. As its best, KNN reaches a training error of 0.36 and testing error of 0.29 after feature selection by correlation and when three nearest neighbors were considered. See Figure 3.

3.4 SVM

With full feature matrix, the lowest training error was zero, which is from nu-SVM with all three different kernels: radial Basis kernel function "Gaussian", Laplacian kernel and ANOVA RBF kernel. Amongst the three kernels, ANOVA RBF had the lowest testing error (0.29). Using reduced feature matrix, the lowest training error rate (0.0) came from nu-SVM classification with Laplacian kernel, although the testing error rate was very high (0.57). The lowest training error rate (0.36) was from bound constraint-SVM classification with hyperbolic tangent kernel, as well as nu-SVM classification with polynomial kernel function. The overall lowest training and testing error was reported in Table 2.

3.5 Logistic Regression

Logistic Regression was implemented on the complete feature matrix created. Table 3 describes the top ten features sorted by most significant p-values. As observed there were very few features that held compelling p-values. In training our LG model the observed training error with LOOCV was 0.44. For the test set our LG model performed worse than our training set, only being about to accurately classify images at random with a test error of 0.5.

3.6 Random Forest

Random forest was implemented in the binary classification of beautiful or not using again the full feature matrix. The training error received was 0.20 on 100 images with LOOCV. The CA and CFA feature variables were observed to play two of the most important roles in classification (See TABLE 4). The results pertaining to the test error was similar to the training error, 0.21. This suggest that our model held a adequate amount of trade-off between variance and bias.

4 Discussion

4.1 Facial Feature Extraction

In order to better understand the Haar Cascades model, a model was developed. While the training algorithm provided by OpenCV [4] was optimized, it is still a very slow algorithm to run. In order to speed up the runtime so iterations on the model could be made, a smaller window and less images were chosen. Given that the algorithm loops over all pixel boxes up to the pixel box size specified (24 x 24 pixels) increasing the pixel size, while likely increasing the accuracy of the model, drastically lengthens training time. A drastic hold-up on training was also the number of negative images. While the algorithm trained instantaneously on the positive images, negative images were processed more slower.

The results showed that testing precision was perfect, meaning that all error was created from the predictor not detecting any feature on a test example. Given that the models and symmetry features depended on an accurate representation of the facial features, it was best to sacrifice some accuracy for higher precision. Thus, the feature detector would only consider a feature correct if there was high confidence in a feature prediction, explaining the high precision.

4.2 Comparison of Classification Models

The observation that Random Forest classifier outperforms Support Vector Machines, Logistic Regression and K-Nearest Neighbor classifiers might be attributed to the presence of noise within the data. Tree classifiers like RF indirectly treat features unequally by discarding irrelevant attributes and using informative/discriminative features more frequently. On the other hand, in SVM classification all features are treated equally and hence SVMs are more sensitive to high dimensional data. Interestingly enough, for the case of Logistic Regression because LR is conceptually simple and has proven to produce robust results, it was expected to perform much better than what was actually observed. Some reasons for this observation could be that LR tries to find broad relationships between independent variables and predicted classes, but without guidance they cannot find non-linear signals or interactions between multiple variables. KNN's low performance, evidently highlights that some features do not contain enough information of the beauty of faces. This would explain why RF out performs KNN in this project. Another drawback of our KNN implementation here is that the best model used a small value of k, k =3. The bias is low due to the small value of k, but the variance is high. [3] This means that our model might not perform well with other data sets. Similarly to KNN, SVM also overfit our data since the best training error was zero, much smaller than the test error. Nu-SVM model worked the best amongst the three models because it puts boundaries on the number of support vectors and errors to deal with the variance-bias trade-off. [8]

Random Forest's high performance in both training and test sets allowed for its clear victory over all other algorithms tested. RF offers a powerful tool in addition to classification, variable importance ranking. This is done by estimating the predictive values of variables by scrambling the variables and seeing how much the model performance drops. When applied here both CA and CFA, which measure overall facial asymmetry, are the two most important feature variables to classification. This correlated well to survey responses, the more beautiful faces were more symmetric according to CFA and FA. Thus, we conclude symmetry can be positively associated with facial beauty.

5 Future Work

Survey results could have been more deeply analyzed to discover unconscious bias that could impact the results. Sexual orientation could also impact a person's perception of beauty. While the survey was pitched as a beauty rating and not an attractiveness rating, the two often go hand in hand so that the attractive person is also deemed beautiful.

With more time to train the Haar Cascade model, a better feature detector could be built using more positive and negative images, and a larger window size. A tighter bounding box would yield a more accurate representation of the face. In fact, a bounding box that is simply a point that identifies separate points on a feature would be preferred. Then, no approximation would be needed to find the center of the feature. Additional features could also be detected to test other features' effect on symmetry and beauty. For instance, Emily Deschanel's face was deemed by the algorithm to be relatively symmetric, but she had a mean beauty score of 4 from survey respondents. Perhaps other features affected people's perception of her beauty. For instance, she has a very strong jawline for a woman. Other detectors could be trained to detect these features as more features would aid in developing a more accurate model. we could do to improve our beauty classifier.

Based on our classifier results, there is a good amount of noise in our feature matrix and only using the reduced feature matrix left out some information. Feature selection could have been done more carefully, i.e. compare various other methods like mutual information or PCA to better represent our data. On a separate note, although the feature matrix was created to best capture the facial feature in our case, there could be other statistics that also contains information about beauty of each face. There were studies that identified points on each face and used pairwise distances between the points as the feature matrix. [6] They further reduced the feature matrix using PCA. In order to quickly collect survey data on the additional 100 images that served as the training set for the model, only binary classifications of beautiful or not beautiful could be gathered. With more allotted time a more precise detector could have been built i.e., a scaled rating rather than binary. Other models like SVR, GLM, EM, and KMeans could also have be evaluated. Lastly, because the models were implemented with default settings, optimizing the model parameters might have improved performance.

6 Bibliography

1. R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.
2. Haar Cascades. URL http://alereimondo.no-ip.org/OpenCV/34
3. Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.
4. OpenCV Github Repository. URL https://github.com/Itseez/opencv
5. Kagian, Amit, et al. "A machine learning predictor of facial attractiveness revealing human-like psychophysical biases." Vision research 48.2 (2008): 235-243.
6. Eisenthal, Yael, Gideon Dror, and Eytan Ruppin. "Facial attractiveness: Beauty and the machine." Neural Computation 18.1 (2006): 119-142.
7. Grammer, Karl, and Randy Thornhill. "Human (< em> Homo sapiens) facial attractiveness and sexual selection: The role of symmetry and averageness." Journal of comparative psychology 108.3 (1994): 233.
8. Hornik, Kurt, David Meyer, and Alexandros Karatzoglou. "Support vector machines in R." Journal of statistical software 15.9 (2006): 1-28.

7 Tables and Figures

Model	Training Error (100 images)	Testing Error (16 celebrities)	Testing Error (116 images)	Testing Precison
Developed Nose Detector	0.80	N/A	0.29	1.00
Online[4] Nose Detector	0.23	0.18	0.07	1.00
Online[4] Eye Detector	0.17	0.30	0.25	1.00
Online[4] Mouth Detector	0.35	0.00	0.07	1.00

Table 1: Error results for the Haar Cascade Feature Detectors

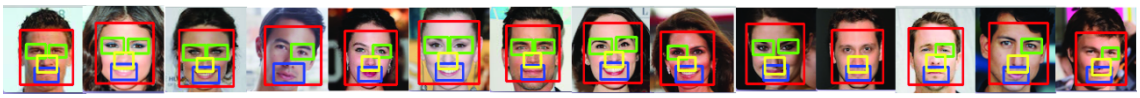

Figure 1: Feature detections made by Haar Cascade model.

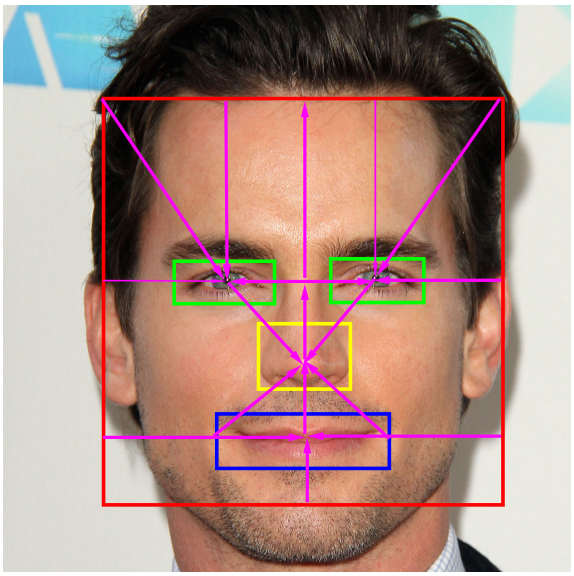

Figure 2: Calculated features represented by purple lines.

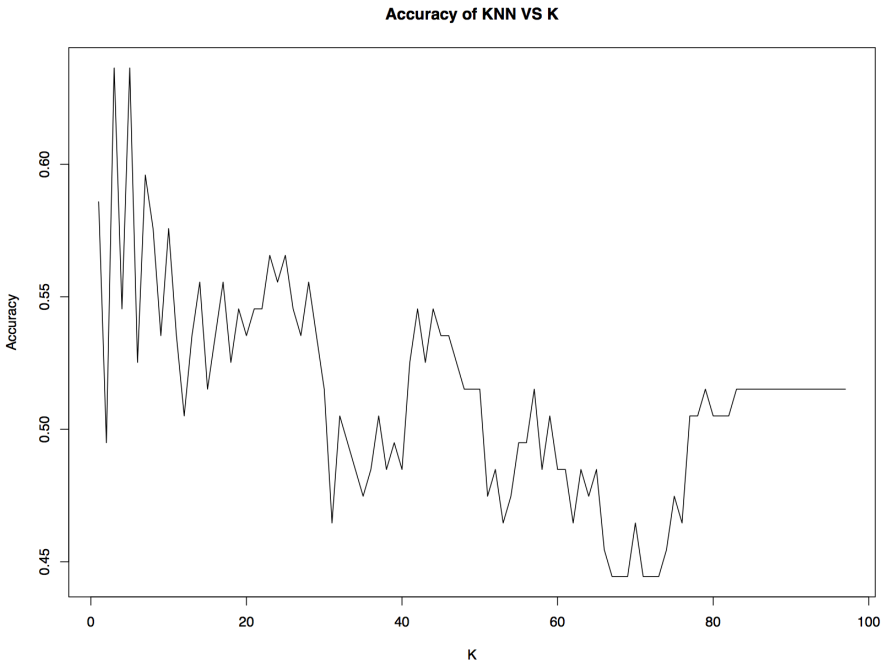

Figure 3: Accuracy of KNN by K Using Whole Feature Matrix in 100 Training Images

Feature	RF Importance Ranking
FA	11.33
CFA	10.78
nose_mouth_r	9.46
forehead_nose_midpoint	9.22
eye_forhead_l	8.19
nose_leye	6.74
diff_pupil_forhead_r	4.48
diff_pupil_face_l	4.44
nose_mouse	3.32
nose_mouth_midpoint	2.95
eye_forhead_r	2.68

Table 4: Top Ten Features from Random Forest Classification Model

Model	Training Error	Testing Error
Random Forest	0.20	0.21
Logistic Regression	0.44	0.50
KNN	0.36	0.29
SVM	0.00	0.29

Table 2: Comparison of Classification Models for Beauty Detection

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	84.92	32.79	2.59	0.01
pupil_face_ratio	-189.67	77.10	-2.46	0.01
nose_mouth_l	-4.26	1.94	-2.20	0.03
diff_nose_face_l	5.34	2.67	2.00	0.05
nose_mouth_r	2.66	1.50	1.77	0.08
d_pupil	2.53	1.44	1.75	0.08
nose_mouth_midpoint	-0.74	0.44	-1.70	0.09
diff_mouth_face_l	-8.32	5.38	-1.55	0.12
diff_pupil_forhead_r	-4.78	3.96	-1.21	0.23
mouth_chin_midpoint	0.57	0.48	1.19	0.23

Table 3: Top Ten Features from Logistic Regression Model