ARE you HOT or NOT®?

CS229 Final Project

Jim Hefner & Roddy Lindsay

1. Introduction

We use machine learning algorithms to predict attractiveness ratings for photos. There is a wealth of psychological evidence indicating that facial beauty is dependent on a number of factors, such as symmetry, averageness, and proportional size of features.[1] It is unclear, however, which of these features would be useful for a machine to learn given a set of two-dimensional images. One application that easily follows from such a project is a web site where a user could upload a photo of themselves to find out how attractive he or she is (or, more accurately, how internet users would rate them) without worrying about potential consequences of having one's photo online. For example, a user could compare several different photos and select the one with the highest score to submit to a match-making website. Previous attempts at using machine learning to classify facial attractiveness have used a very regular set of images, and ratings compiled from votes by the same group of people.[2] We attempt to predict scores for highly variable photos.

2. Materials

Our dataset comes from the web site HotOrNot.com, which hosts photos submitted by users to be rated by the public. A person's "hotness" is rated on a scale of one to ten, rounded to the nearest tenth. We focused our efforts exclusively on photos of females, and we came across photos rated as high as 9.9 and as low as 3.2, with vote totals ranging up to several thousand. Training images were filtered based on the number of votes and the positioning of the subject. We used 100 votes as the minimum number required to include the photo, since we found personal estimations of attractiveness to be highly correlated with the given ratings with that many votes. We chose photos with women facing forwards with an unblocked face, at a resolution that would make the face still discernable when scaled to 100 x 100 pixels. Photos that featured too much of the subjects body (not just the face) were eliminated to prevent bias unrelated to facial attractiveness. A total of 197 photos were collected in this manner. The dataset was then split into a training set of size 147 and a test set of size 50.

3. Eigenfaces

3.1. Eigenfaces—Methods

The first approach employs Principle Component Analysis. We used the eigenfaces[3] method for image processing and adapted code for the MATLAB® implementation.[4] Following collection of the dataset, each photo was cropped to a square which started just above the left eye and extended to the analogous point above the right eye. To counteract the vast variability in lighting, each photo was embossed using a 3-pixel convolution filter. This filter is known to improve machine-learned facial recognition under variable lighting conditions.[5] Finally the images were converted to grayscale and resized to 100x100 pixels.

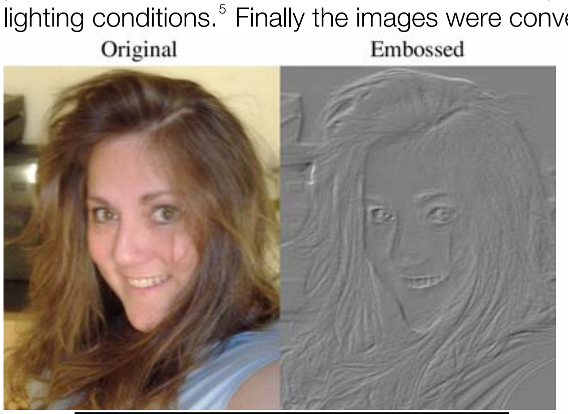

Fig. 3.1.1: An example of the embossing procedure.

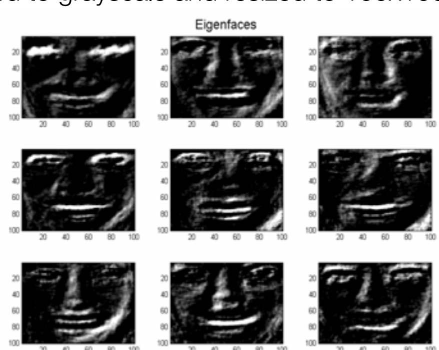

Fig. 3.1.2: A subset of the resulting eigenfaces.

---

[1] Zimmer, 2001
[2] Eisenthal et al., 2006
[3] Turk and Pentland, 1991
[4] Serrano
[5] Kouzani et al., 1998

We used correlation between eigenfaces features and attractiveness ratings to find the best eigenfaces to use as features for the learning algorithm. The maximum correlation between the coefficients of an eigenface for the training data and the training scores was just over 0.2. Expressing each face as a linear combination of eigenfaces yields a highly variant and non-linear dataset, and attempts at score estimation using regression methods were exceedingly inaccurate. We elected to classify the data using a support vector machine, and we used k-Nearest Neighbors as a baseline for comparison. The training data was separated into "hot" and "not hot" categories, with the threshold of 7.5 separating the categories (the mean of the training set was approximately 7.5, and the median was 7.6).

To determine the number of neighbors to use for the kNN and the number of eigenfaces to use as features for the SVM, we used a forward search feature selection wrapper model. Features were added in decreasing order of attractiveness correlation, from three to sixteen eigenfaces and the appropriate training algorithm was then retrained using those k features. Cross-validation was used to determine accuracy, training and testing on the sets as described above.

The SVM was implemented via Platt's SMO algorithm[6], with tolerance = 0.001, C=1, and max_passes = 10. The input features were the coefficients for each photo of the most highly correlated eigenfaces and the target variables were contained in a vector of 1's and -1's, corresponding to whether an input image represented a face with score >= 7.5 or < 7.5, respectively.

3.2 Eigenfaces—Results

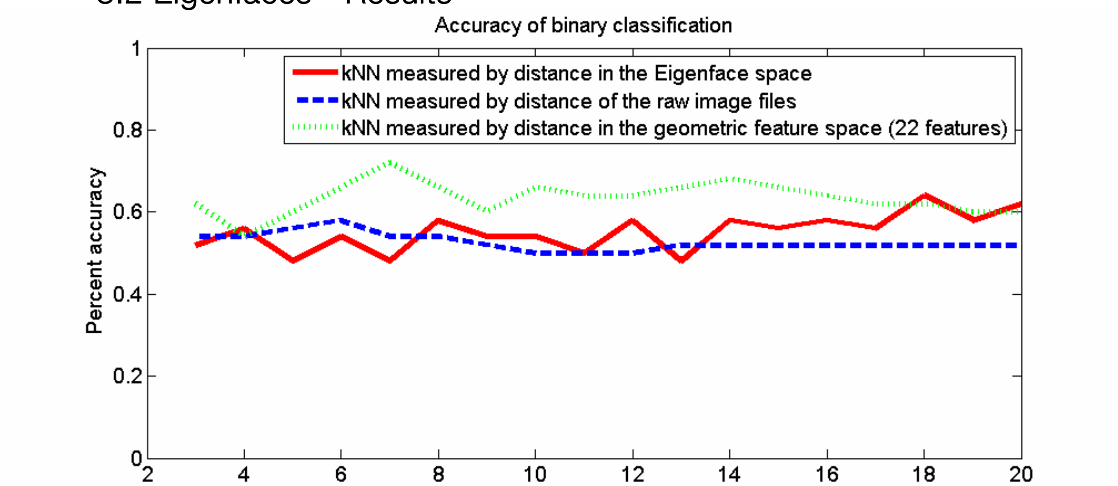

Fig. 3.2.1: Accuracy of kNN when computed using vectors composed of Eigenface coefficients, raw pixel intensity, and facial geometry distance ratios.

k-Nearest Neighbors was implemented, calculating neighbors both by Euclidian distance of the pixel intensities of the raw image files and of the feature space for the sixteen most highly correlated eigenfaces. We calculated the kNN accuracy for every number of neighbors between one and twenty, although due to ambiguity (i.e. an equal number of "hot" and "not hot" neighbors) only the odd numbers of neighbors are statistically relevant. For the Euclidian distance of the raw image files, the kNN classifier was essentially at chance (50-56% accuracy), with maximum performance at five neighbors (*Fig. 3.2.1*). For the Euclidian distance in the eigenface feature space, accuracy ranged from 46% to 58% depending on the number of neighbors used (*Fig. 3.2.1*). While the SVM did converge, it failed to make any semblance of accurate predictions on the test data. The maximum accuracy was 52% and the minimum was 48%, i.e. at chance.

4. Facial Geometry

4.1 Facial Geometry—Methods

The second approach makes predictions based on facial geometry (the ratio of the distances between various points on the face). For the preprocessing, the original photos were cropped about the face as before to remove irrelevant background. Next, a MATLAB® script was written which cycled through

[6] Platt 1998

the images and displayed them on the screen. Using the `ginput()` function, each photo was tagged with a series of 19 points believed to be related to attractiveness (*Fig 4.1.1)*.

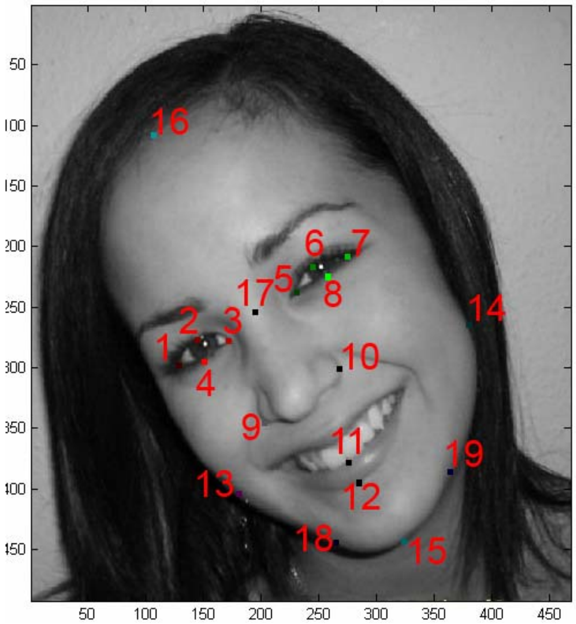

Fig 4.1.1: An example face tagged with the ordered points used in the facial geometry method.

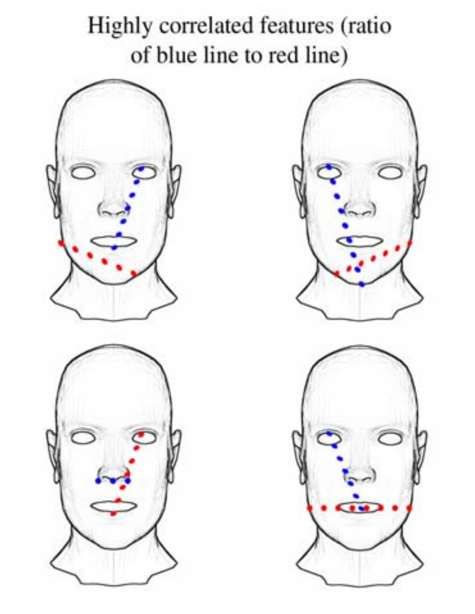

Fig 4.1.2: Several of the features found to be most highly correlated with facial attractiveness.

Rather than select our input features to be geometric ratios based on preconceived notions of facial attractiveness, the ratios most highly correlated with attractiveness were selected. This was done via brute force: an algorithm was implemented which computed the ratio of the distance between every two-point subset to the distance between every other two-point subset. Next, correlation was computed between each of these ratios and the given scores on the training set. Several of the most highly correlated features are displayed above (*Fig 4.1.2*). The maximum correlation coefficient was 0.44. We chose eleven of the top ratios which corresponded a diverse set of facial geometries, and their counterpart on the opposite side of the face. For example, we would discard a ratio that included the top of the eye to the chin if the similar ratio of the bottom the eye to the chin was already chosen as a feature.

As a baseline, k-Nearest Neighbors was implemented measuring Euclidian distance of 22 facial geometry features. The support vector machine, using the SMO algorithm, was then implemented for binary classification. We also implemented linear regression for score estimation. For all three algorithms, we used the forward search feature selection wrapper method to find the optimal number of features or neighbors to use.

4.2 Facial Geometry—Results

Results for the facial geometry method were a vast improvement over the eigenfaces method. kNN classified test images with a maximum of 72% accuracy using seven neighbors, and with a minimum of 60% accuracy using nineteen neighbors (*Fig. 3.2.1)*. The SMO algorithm achieved a maximum training set accuracy of 68% at 8, 15, 17, 18, 20, & 22 features used (*Fig. 4.2.2),* which is well above chance. For linear regression, the maximum correlation between the predicted scores and the actual scores for the test data was 0.59, using seventeen ratios as features (*Fig 4.2.1)*.

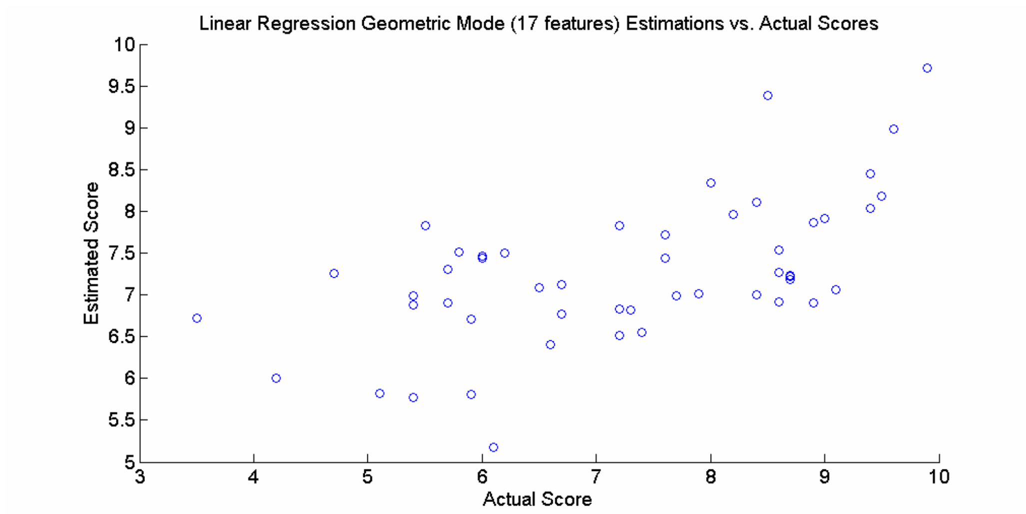

Fig. 4.2.1: Predicted decimal scores vs. actual scores using linear regression.

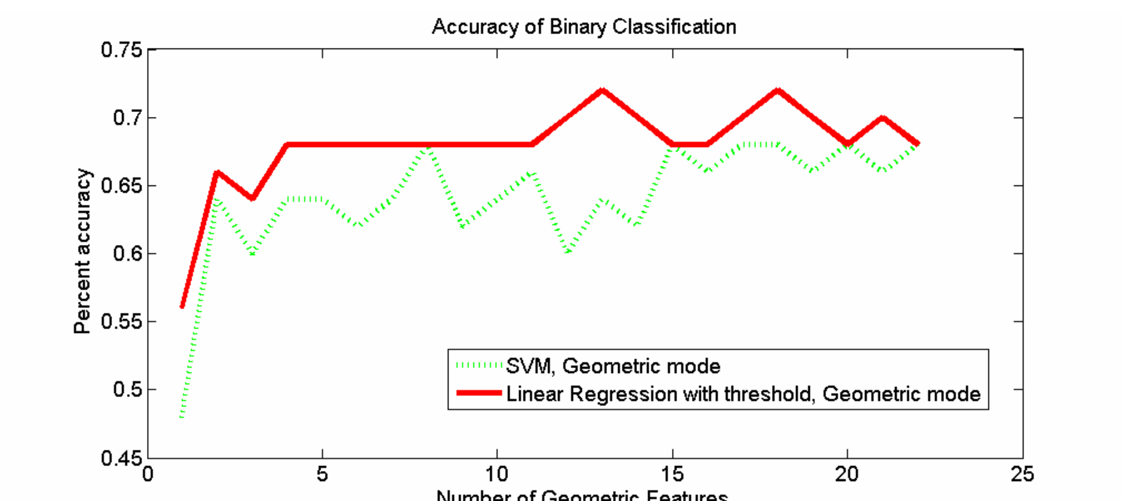

Fig. 4.2.2: Binary classification accuracy vs. number of facial features used for both SVM and linear regression with threshold.

5. Discussion

With our training data, the facial geometry method was more successful in predicting facial attractiveness than the eigenfaces method. The data was drawn from a highly irregular set of images; slightly different head positions, facial expressions, facial blockage due to hair and glasses, and picture resolutions greatly impacted the eigenfaces while effectively preserving facial geometry information. With such varying images and a limited training set size, the eigenfaces generated simply were not good enough to accurately reconstruct the input images. Applying the embossing filter to all of the images in the dataset had the positive effect of eliminating the lighting discrepancies, but it may also have eliminated facial features that are important indicators of attractiveness. The overall low correlations between individual eigenfaces and attractiveness indicate that a much larger training set would be necessary to utilize the eigenfaces approach, since the data is so non-linear. Given these low correlations, the fact that the SVM and the kNN did not score much higher than chance is not surprising.

We believe the facial geometry method performed well primarily due to the method of feature selection. The results of the brute force method to select relevant features were surprising. Whereas psychological research indicates that humans calculate attractiveness of conspecifics using pure ratios such as nose width to height, eye to nose distance, and chin to jaw distance, these particular features were found to be poorly correlated with attractiveness in our dataset. The 0.60 correlation for linear regression estimation is comparable to the 0.65 correlation obtained by *Eisenthal et al.* that used a highly controlled dataset trained on the “psychological” features.

6. Conclusion

Our results lead us to believe that facial attractiveness can be predicted fairly accurately with machine learning algorithms on a diverse training set of two-dimensional images. We found the facial geometry approach to be superior to the eigenface approach. 0ur predicted to actual score correlation of 0.60 is comparable to that of *Eisenthal et al.* with a correlation of 0.65. We believe our success on a diverse training set is due to the fact that we did not pre-select the training features based on psychological evidence, but found novel and unpredicted features by the brute-force method. Further research should determine the upper boundary on score correlation by increasing the size of the training set and further optimizing the feature selection of facial geometry.

7. References

[1] Eisenthal, Yael, Gideon Dror, and Eytan Ruppin. "Facial Attractiveness: Beauty and the Machine." Neural Computation (2006): 119-142. 18 Nov. 2006 <http://portal.acm.org/ft_gateway.cfm?id=1117680&type=pdf&coll=&dl=acm&CFID=15151515&CFTOKEN=6184618>

[2] Kouzani, A Z., F He, K Sammut, and A Bouzerdom. "Illumination Invariant Face Recognition." Ieeexplore.Org. 1998. Edith Cowan University. 17 Nov. 2006 <http://ieeexplore.ieee.org/iel4/5875/15682/00727511.pdf?arnumber=727511>.

[3] Platt, John. Fast Training of Support Vector Machines using Sequential Minimal Optimization, in Advances in Kernel Methods – Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds., MIT Press (1998).

[4] Serrano, Santiago. "Eigenface Tutorial." Drexel University. Drexel Santiago. 16 Nov. 2006 <www.pages.drexel.edu/~sis26/Eigenface%20Tutorial.htm>.

[5] Turk, M. and A. Pentland (1991). "Face recognition using eigenfaces". Proc. IEEE Conference on Computer Vision and Pattern Recognition, 586–591

[6] Zimmer, A. "BEAUTYCHECK." 2001. Universitat Regensberg. 15 Nov. 2006 <http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Psy_II/beautycheck/english/bericht/beauty_mi_zensiert.pdf>.