AMATH 563 HW 3
Clustering and Classification of Yale Faces

Katie Johnston

May 27, 2020

Abstract

In this project, we explored the Yale Faces data set. First we explored the cropped and uncropped images using SVD analysis and found that there are a few dominant modes but it takes about rank 100 for a good low rank approximation. Next, we classified the data set using supervised learning methods and found that linear discrimination analysis performed the best across our classifications tested. We also explored some unsupervised learning methods on our data and discovered that the groups were mostly chosen by the lighting of the images.

1 Introduction and Overview

In the project, we explored the Yale Faces data set. First, we explore the cropped and uncropped images through an SVD analysis. This allows us to learn about our system and compared the differences between the cropped and uncropped images. Next, we explored the cropped images through classification and clustering methods. We explored the data set through supervised and unsupervised learning methods. For classification we explored identifying a single individual from the group, each individual in the group together, and males and females. Classification has a lot of applications in a large variety of fields and this exploration on this data set helped us learn some of the power of the popular methods.

2 Theoretical Background

In the first part of this project, we did performed an Singular Value Decomposition (SVD) analysis of the Yale faces both cropped and uncropped. The equation for the SVD is given by

$$A = U\Sigma V^T$$

From an SVD of a system like this, we can learned quite a bit about our data. From the singular values, we can learn how many important modes there are and what percentage of the information they represent. From the U matrix, we can determine what the most important modes look like, and from the V matrix, we can find the weighting between the modes for given images. We can also make a low rank approximation to our data, using the formula,

$$A_n = \sum_{i=1}^{n} \sigma_i U_i V_i^T$$

This low rank approximation will enable us have a smaller representation of our system while still containing almost all of the information from the images.

Next, we want to classify our images based on a number of factors. There are a large number of supervised learning algorithms which are good at this type of task. We started with supervised learning because we could specify the labels, and the resulting predictions had the same interpretation as these labels. Some methods used include K-Nearest Neighbors, Naive Bayes, Linear Discriminant Analysis, Decision Trees, and Support Vector Machines. Here is a short description of the ideas behind these algorithms:

- **k-nearest neighbors** looks at the k (odd number) neighbors of a point, and classifies the point as the type of the most neighbors. You need an odd number to avoid ties.
- **Naive Bayes** is based on Bayes' theorem, and utilizes conditional expectation. It estimates the label of the new point based on the prior distribution of the labeled points.
- **Linear Discriminant Analysis** separates the data into two or more groups using a number of lines.
- **Decision Trees** divide by the data by different conditions every step until a tree is formed.
- **Support Vector Machines** works by projecting the data onto higher dimensional spaces and then splitting the data with hyperplanes

We can also learn a lot from our data using unsupervised methods. Unlike supervised learning, unsupervised learning does not have labels associated with the data points. Thus, it is harder to find the meaning of the groups that these unsupervised learning algorithms produce. The methods we tried in this project were k-means clustering and Gaussian Mixture models.

3 Algorithm Implementation and Development

First we imported our data from the image files and formed matrices where each column was a single image from the file. For example, the cropped faces matrix was $(2432, 32256)$ because there were 2432 images, and each image had a total of 32256 pixels. For the cropped images, we also made a matrix that was sorted with females in the first 512 images and males in the remainder.

3.1 SVD Analysis

For the SVD Analysis, we started by taking the SVD of the cropped and uncropped images. We plotted the normalized singular values, the first some columns of U, and looked at the columns of V. We also made low rank approximations by taking the first r rows of U, singular values, and first r rows of V. We were also able to find the amount that each approximation held by summing the first r singular values.

3.2 Classification and Clustering

Next for the classification and clustering, we only focused on the cropped images. We tried 3 different classification problems on a variety of supervised learning methods. First, identifying the first person, so we made a labels that were 1 if it were the first person and 0 otherwise. Next, we looked at categorizing each individual, so our labels vector contained the numbers 0-37 depending on which individual number is was. And finally, we classified males and females so our labels vector contained a 1 for female and a 0 otherwise.

In order to cross validate, we first broke up our data set into training and testing sets, where the training was 80% of the data and the testing was the remaining 20%. These sets were randomly chosen across the dataset.

We performed a variety of supervised learning methods on our data sets, including Least Squares Regression, Lasso, k-Nearest Neighbors, Navie Bayes, Linear Discriminant Analysis, Quadratic Discriminant Analysis, Support Vector Machines, and Decision Trees. Python packages were used for all these and further description can be found in Appendix A.

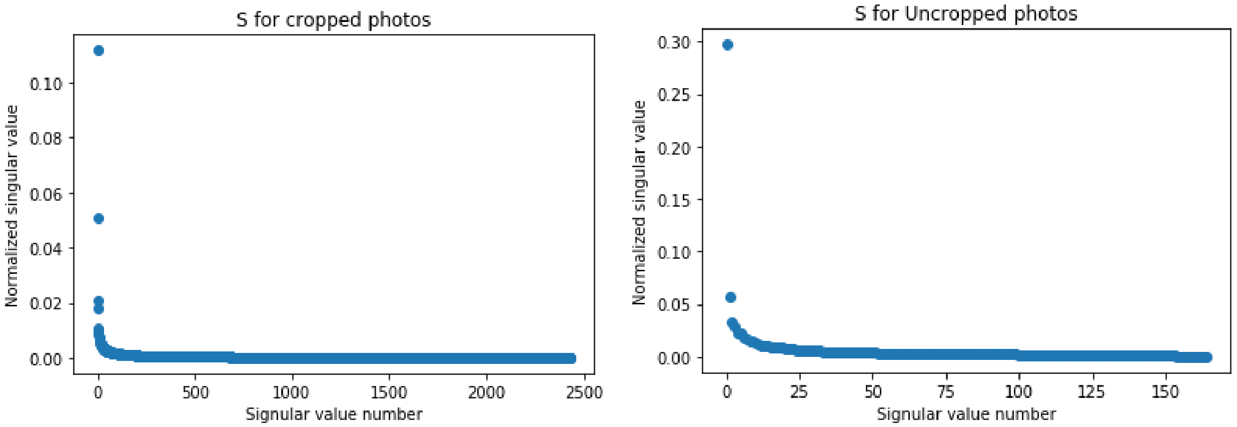


Figure 1: Singular values for Cropped and uncropped data

To calculated error we calculated the accuracy of each algorithm, which was the percentage of points classified correctly. We calculated the accuracy on the training and testing set, and the algorithm with the highest testing accuracy was determined to be the ”best” model.

For the unsupervised clustering, we used k-means and Gaussian Mixture Models. For k-means, we plotted which images were in which group, divided by person and image number. For the Gaussian Mixture Models, we compared the groups found using k-means when k = 2 to the clusters found by Gaussian Mixture Models.

4 Computational Results

4.1 SVD Analysis

First we compared the cropped and uncropped images. In Figure (1), you can see the normalized singular values for the cropped and uncropped images. You can see that both have one most important mode and a few more less important but significant modes. In Figure (2), you can see a few of these modes, which are columns of U. For the cropped images, you can clearly see a face for the first mode. When you can only see the outline of a face with some slight variation for the eyes and mouth for the uncropped mode. For the cropped images, we are seeing more of the facial structures, when for the uncropped images, we mostly see the position of the head, and less of the actual face.

In Figure (3), you can see the first three columns of V for the cropped images sorted by male (blue) and female (orange). You can see that the female images are more on the top part if the ball like structure. This shows that there are differences between the weighting in males vs. females, but it is not linear in the first three modes. It may be as we add more modes.

We also looked at the rank of the images. In Figure (4), you can see the low rank approximations of the first images in the cropped and uncropped photos. You can see that by rank 100, both images are identifiable, still a little blurry, but you can identify the person in the image. For the uncropped images, 91.5% is represented by this 100th mode. However, only 61.9% is for the cropped photo. We can also observe this behavior by the values of S shown in Fig (1) because there are more large singular values for the cropped images, and the large singular values for the cropped images are much smaller than the large singular values for the uncropped images, meaning that more information is presented in the first few modes of the uncropped images than the cropped ones.

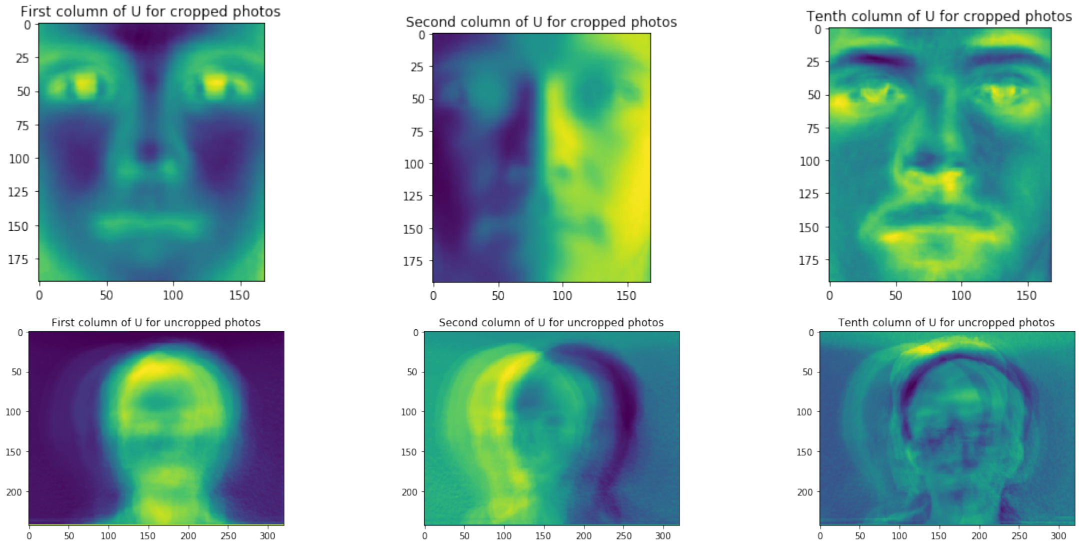


Figure 2: Columns of U: First row show for cropped data, and Second row show for uncropped data.

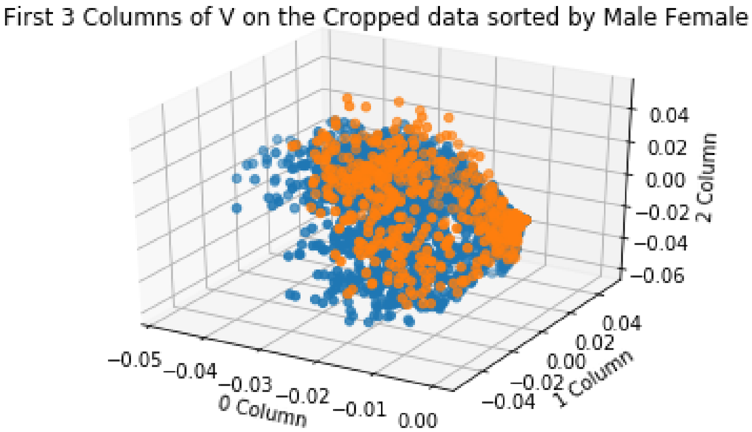


Figure 3: 3D view of the first 3 columns of V for the cropped photos separated by Female (orange) and Male (blue)

4.2 Classification

Next, we want to be able to classify people in the images. We looked at classifying a single individual, all the people separately, and male vs. female. We used a variety of supervised learning methods to build these classifiers. Figure (5a) shows the training and testing error when trying to choose just the first person in the data set. Figure (5b) shows the error when trying to classify each individual, and Figure (6) shows the error when trying to distinguish between male and female. From these graphs, you can see that each situation choose a different ”best” method. You can also see that some methods had very high training accuracy (possible at 1.0), yet very low testing accuracy. This shows that some of these algorithms are overfitting our data. Across all these classifying situations, it looks like Linear Discriminant Analysis performs the best.

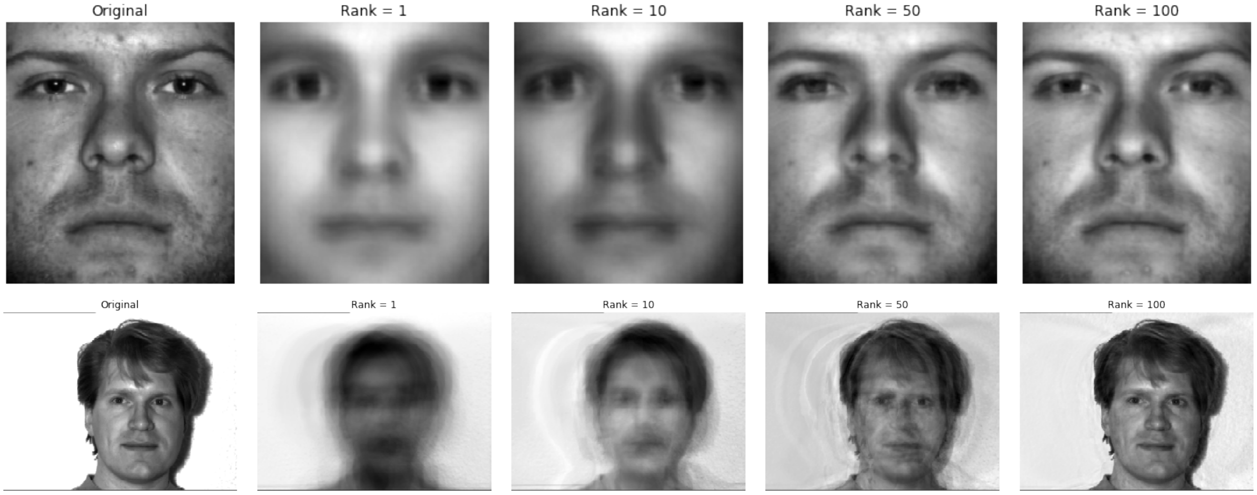
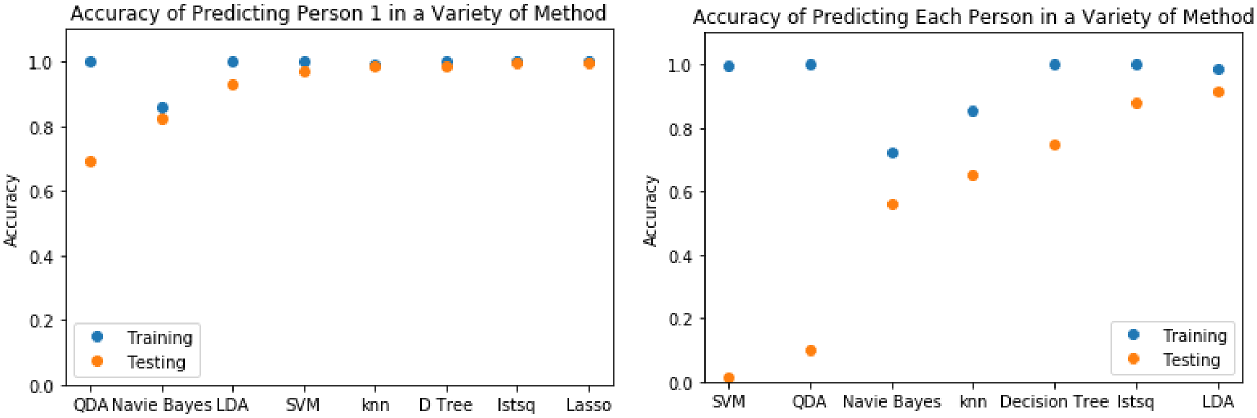


Figure 4: Low Rank Approximations with increasing Rank. Top row is the cropped images and bottom row is the uncropped images



(a) Predicting the first person's face using a variety of methods

(b) Predicting the each person's face from the group using a variety of methods

Figure 5

4.3 Unsupervised Learning

Next, we tried our data set on some unsupervised learning algorithms. The results from the k-means algorithm with $k = 2, 3$ can be seen in Figure(7). These plots have the photo number of each individual on the x-axis and the group on the y-axis. The different colors represent different people, and we had to offset them to be able to view all the dots. From these plots, you can see that the algorithm is choosing groups based on the lightning, which is the difference between the different pictures of all the individuals. Group 0 for the 2-means is the more well lit groups, and Group 1 is the darker groups. Part of this grouping could also be that in the darker photos, all the face features are harder to see, so that might be what the algorithm is picking up on. The algorithm was definitely not grouping by gender, as the accuracy on gender was 47.5%, which is better than guessing (21% are female), but still not great.

We also looked at Gaussian Mixture Models. We attempted to find two groups that were Gaussian distributed for the cropped data set. You can see some of our results in Figure (8). The coloring of

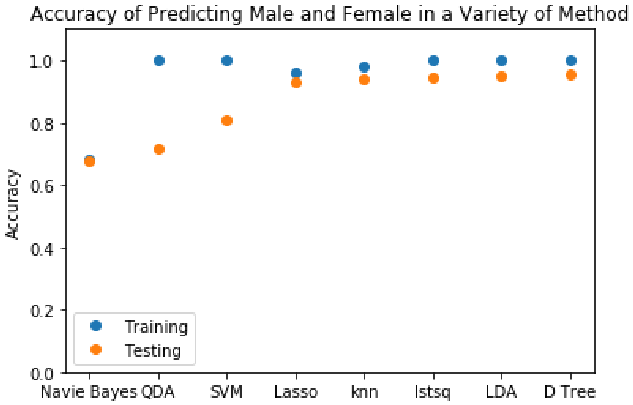


Figure 6: Predicting Male vs Female using a variety of methods

these plots show the groups 0 and 1 of k-means clustering with $k = 2$. You can see from these plots that the groups found by k-means is similar to the Mixture Models, yet have some differences because there is a lot of overlapping red and blue points. From this grouping comparision, we can see that the Mixture models is also grouping more by lighting than by gender.

5 Summary and Conclusions

In this project, we found a number of interesting results. In our SVD analysis, we found that there is one very dominant mode, and a few also significant modes in both the cropped and uncropped modes. But that it's at about rank 100 that the low rank approximation has presents an identifiable person. We also see that the more important mode of the cropped images looks like a person's face with it's features, while the uncropped images just looks like the outline of a head. We also observed that in the first three columns of the V matrix, the female and male groups looks most separable, but definitely not linearly.

Next, we looked at classification of our cropped images. For our three classification problems, we found that each one chose a different "best" model, which had the highest accuracy on a testing set, but overall LDA performed the best across. We also observed that some of these methods overfit our data and that is why cross-validation is so important.

Lastly, we clustered our data using two unsupervised learning methods. We found that the groupings were not by gendered, but had more to do with the lighting of the images. We also found that when we made 2 groups with our two methods, the groups were similar but not exactly the same.

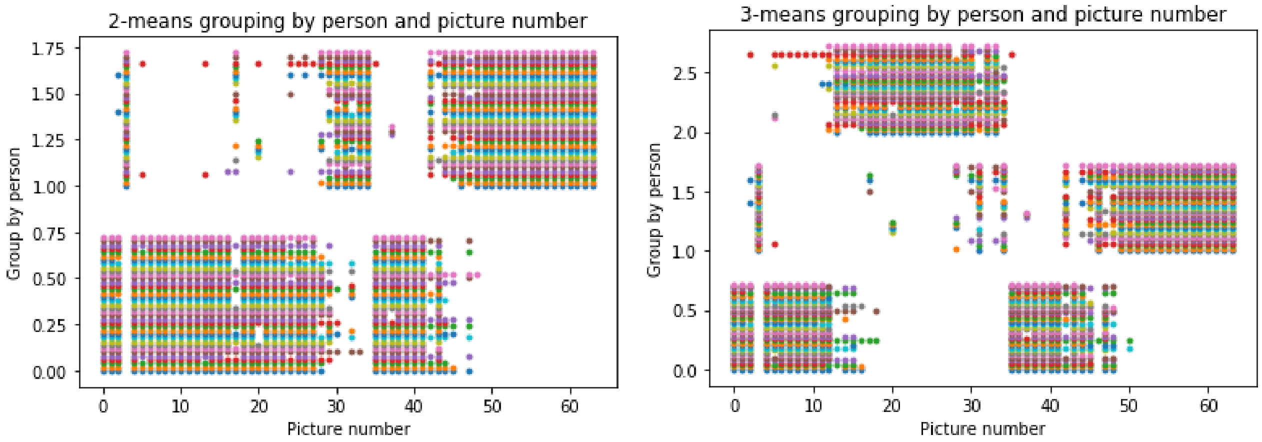


Figure 7: Grouping faces using k-means

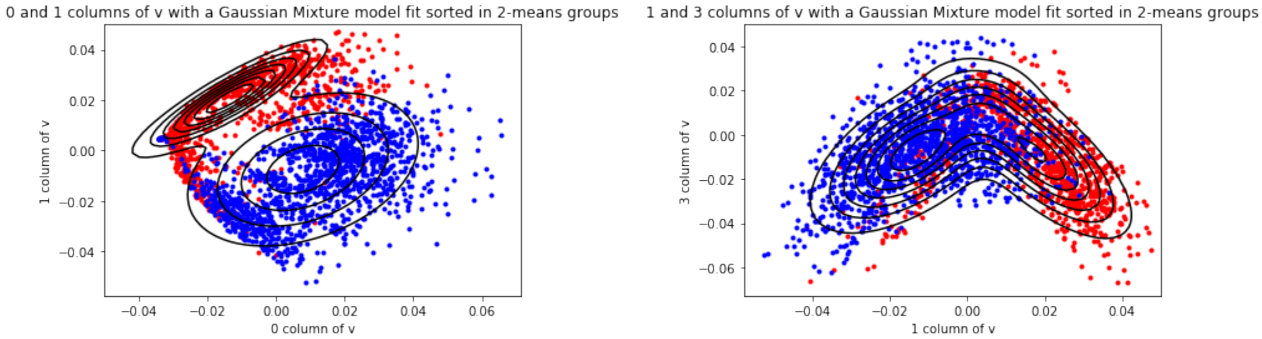


Figure 8: Grouping faces using Gaussian Mixture Models. Grouping into two groups, and the colors represent the two groups found using 2-means clustering

A Python functions used and brief implementation explanation

A large number of python package functions were used, including:

- `np.linalg.lstsq` the least squares fit from numpy's linear algebra library.
- `sklearn.linear_model.Lasso` sklearn's Lasso method
- `sklearn.neighbors.KNeighborsClassifier` sklearn's k nearest neighbor classifier
- `sklearn.naive_bayes.GaussianNB` sklearn's Gaussian Naive Bayes method
- `sklearn.discriminant_analysis.LinearDiscriminantAnalysis` sklearn's LDA classifier
- `sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis` sklearn's QDA classifier
- `sklearn.svm` sklearn's svm method
- `sklearn.tree.DecisionTreeClassifier` sklearn's decision tree classifier
- `sklearn.cluster.KMeans` sklearn's k-means cluster algorithm
- `sklearn.mixture.GaussianMixture` sklearn's Gaussian Mixture model algorithm

  To use this, first had to find the SVD of our data set and then enter the desired number of columns of v into this method

B Python Code

Please see Github for code: `https://github.com/KatieJ16/AMATH563/tree/master/HW3`

Notebooks included:

- `HW3-SVD.ipynb` Contains code relevant to the SD Analysis
- `HW3-One-Face.ipynb` Contains code relevant classifying just one face
- `HW3-All-Faces.ipynb` Contains code relevant classifying all faces at once
- `HW3-Male-Female.ipynb` Contains code relevant classifying males and females
- `HW3-unsupervised.ipynb` Contains code relevant clustering using unsupervised learning methods