AMATH 563 Homework 3

Daniel Burnham (https://github.com/burnhamdr)

May 18, 2020

Abstract

The work presented here is motivated by material covered in AMATH 563 Inferring Structure of Complex Systems regarding the applications of Singular Value Decomposition (SVD) and classification algorithms. This report will discuss these topics in two sections. The first section will report on the application of SVD to a data set of human faces. SVD is a linear algebra matrix factorization method that represents how the matrix of interest stretches/compresses and rotates a given set of vectors. The SVD is a powerful tool because the constitutive components of its factorization allow for the subsequent projection of a matrix onto low-dimensional representations. In the problem addressed here this is used to determine the rank (dimensionality) of a data set of human faces. The results demonstrate that faces can be sufficiently represented with low rank approximations when the face images are cropped in a standardized manner. In applying SVD to uncropped faces, the dimensionality reduction is less effective in creating a low rank approximation of individual faces. The second section will report on the implementation of various classification algorithms for features of the cropped face images. K-Nearest Neighbors, Naive Bayes, Linear Discriminant Analysis, Support Vector Machine, and Decision Tree classifiers are utilized to discriminate each face from the others and also to classify faces based on gender. The work presented in this report describes fundamental strategies to address modern data science problems.

1 Introduction and Overview

The topics explored here are discussed in two sections. The first section involves the analysis of both cropped and uncropped face image data using SVD. The second section involves face identity and gender classification algorithms.

1.1 SVD Face Data Analysis

Two data sets were analyzed with SVD: cropped face images, and uncropped face images. SVD analysis was performed on both of these data sets to investigate how many modes are necessary for good image reconstruction (i.e. determine the rank of the face space). Differences in the singular value spectrum between the two data sets were subsequently identified and are discussed further in the Computational Results section of this report.

1.2 Face Classification

In keeping with the theme of data feature extraction, characteristics of cropped face image data are leveraged to classify face identities and face genders in this part of the report. Identity and features delineating gender are seemingly easily recognizable to the human eye, which raises the question of how these distinctions are drawn? The objective of the work in this section of the report is to implement computational algorithms that capably classify cropped face images. This objective was pursued in the context of three specific tasks:

1. (test 1) Face Classification: From the various faces build classifiers that can reasonably identify an individual face.

2. (test 2) Gender Classification: The above experiment was repeated, but with only two classes (one for each gender). These gender classifications were labelled by hand through subjective observation and therefore may not accurately represent the gender identities of the individual faces.

3. (test 3) Unsupervised Algorithms: Expanding upon the results of the first two tests, clustering methods were employed to find patters in the faces that naturally cluster.

Each of these test exercises contributed to illustrating the ability of computational methods to distinguish characteristics of face images. Understanding these methods perhaps can shed light on possible methods used by the brain for similar tasks.

2 Theoretical Background

2.1 Singular Value Decomposition (SVD)

SVD is a linear algebra method for factorization of a matrix into a number of constitutive components. It is rooted in the observation that during matrix vector multiplication of a vector $\mathbf{x}$ by a matrix $\mathbf{A}$ the resulting vector $\mathbf{y}$ has a new magnitude and direction. This transformation of the vector $\mathbf{x}$ by a matrix $\mathbf{A}$ implies that perhaps the action of matrix $\mathbf{A}$ can be replicated through component matrices that perform the same magnitude and direction manipulations. Furthermore, it would be beneficial if these components possessed properties that made them easy to work with, such as orthogonality and diagonality. The SVD factorization of the matrix $\mathbf{A}$ achieves these goals by expanding this observation of vector transformation under multiplication by a matrix to $\mathbb{R}^m$. SVD does this by building from the observation that the image of a unit sphere under any m×n matrix is a hyperellipse. Following from this, one can represent this transformation as $\mathbf{A}\mathbf{v}_j = \sigma_j \mathbf{u}_j$ for $1 \leq j \leq n$. Where $\mathbf{v}_j$ are the vectors of the unit sphere transformed by $\mathbf{A}$, and $\sigma_j \mathbf{u}_j$ are the resulting transformations representing the the semiaxes of the hyperellipse. Rearranging this equation allows for the SVD factorization of $\mathbf{A}$ to be written as follows (Kutz 2013):

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{1}$$

In this form, $\mathbf{U}$ is unitary and contains the vectors $\mathbf{u}_j$ indicating the directions of the transformed hyperellipse semiaxes, $\mathbf{\Sigma}$ is diagonal and contains the scaling values corresponding to these semiaxes, and $\mathbf{V}^*$ is the Hermitian transpose of $\mathbf{V}$ which contains the orthonormal basis of the vectors that are transformed under $\mathbf{A}$.

It is worth noting that it can be proved that every matrix $\mathbf{A} \in \mathbb{C}^{m \times n}$ has a singular value decomposition and the singular values are uniquely determined (Kutz 2013). Additionally, if the matrix $\mathbf{A}$ is square, the singular vectors $\mathbf{u}_j$ and $\mathbf{v}_j$ are also uniquely determined up to complex signs (Kutz 2013). This is significant because it allows for every $\mathbf{A} \in \mathbb{C}^{m \times n}$ to be factorized by SVD and subsequently represented with lower rank matrix approximations. This will be useful in the context of the face image data set explored here as it represents a way of reducing the dimensionality of the face feature space.

2.2 Classification Methods

2.2.1 K-nearest Neighbors

The K-nearest Neighbors classification method aims to classify data points by determining the proximity of the data point in the data space to a predefined number of training samples. The proximity is then used to classify the data point by a plurality vote. The number of clusters to create in the training set is set by the implementer. The algorithm is simple in nature compared to others profiled in this work, but has achieved success in many problem contexts especially in situations that have irregular decision boundaries.

2.2.2 Naive Bayes

This is a supervised algorithm that is based upon Bayes' rule. Suppose two classes of data, 0 and 1, need to be separated. The following ratio is constructed:

$$\frac{P(1|x)}{P(0|x)} \tag{2}$$

where $P(1|x)$ is the probability of having class 1 given the data x and $P(0|x)$ is the probability of having class 0 given the data x. This can be rewritten as:

$$\frac{P(1|x)}{P(0|x)} = \frac{P(x|1)P(1)}{P(x|0)P(0)} \tag{3}$$

Using the ratio, we can get a metric that provides a way of separating the data. In our case, the Gaussian Naive Bayes classifier is used where we assume a Gaussian probability distribution.

2.2.3 Linear Discriminant Analysis

The goal of Linear Discriminant Analysis (LDA) is to find the best projection of the data that maximize class separation (see Figure 1). This is achieved by mathematically constructing the optimal projection basis which separates the data (Kutz 2013). In the case of classification of two distinct classes with LDA, a projection w is constructed as:

$$\mathbf{w} = \arg\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \tag{4}$$

Where $\mathbf{S}_B$ and $\mathbf{S}_W$ are referred to as the scatter matrices for inter-class ($\mathbf{S}_B$) and intra-class ($\mathbf{S}_W$) data are mathematically derived as:

$$\mathbf{S}_B = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T \quad \mathbf{S}_W = \sum_{j=1}^{2} \sum_{\mathbf{x}} (\mathbf{x} - \mu_j)(\mathbf{x} - \mu_j)^T \tag{5}$$

The μ terms are arrays of the average value for each class. The scatter matrices capture the variance within each of the data classes as well as the variance of the difference in the means between classes. From these scatter matrices Equation 4 can be solved by way of solving the eigenvalue problem

$$\mathbf{S}_B = \lambda \mathbf{S}_W \mathbf{w} \tag{6}$$

The eigen vector of the maximum eigenvalue calculated from Equation 6 is the projection basis for LDA to maximally separate the two classes.

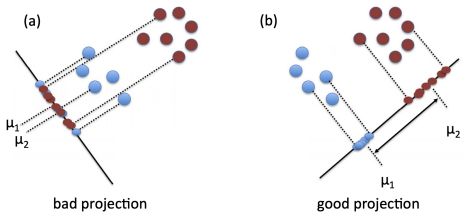

Figure 1: Two-class LDA task where an appropriate projection basis (b) is shown versus a poor performing projection bases (a). (Kutz 2013)

2.2.4 Support Vector Machine

The goal of the Support Vector Machine algorithm is to identifying a hyperplane that optimally separates data points of distinct classes. The optimal hyperplane is one that maximizes the distance (margin) from data points of either class to the hyperplane. The general equation for a hyperplane is given by (Brunton 2019):

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{7}$$

where the vector w and constant b are the parameters that are optimized. Given the hyperplane a data point $\mathbf{x}_j$ can be classified by simply computing the sign of $\mathbf{w} \cdot \mathbf{x}_j + b$. The loss function for the optimization problem can thus be formulated as (Brunton 2019):

$$\ell(\mathbf{y}_j, \bar{\mathbf{y}}_j) = \ell(\mathbf{y}_j, sign(\mathbf{w} \cdot \mathbf{x}_j + b)) \tag{8}$$

To frame this loss function as also maximizing the margin between the data points and the hyperplane and not simply the correct classification (Brunton 2019):

$$\arg\max_{\mathbf{w},b} \sum_{j=1}^{m} \ell(\mathbf{y}_j, \bar{\mathbf{y}}_j) + \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } \min_j |\mathbf{x}_j \cdot \mathbf{w}| = 1 \tag{9}$$

In practice a Hinge loss function is employed as it is a smooth function that counts the number of errors in a linear way therefore allowing for piecewise differentiation (Brunton 2019). This method can be adapted to nonlinear class separations mapping the data into a nonlinear, higher-dimensional space (i.e. $\mathbf{x} \mapsto \Phi(\mathbf{x})$). $\Phi(\mathbf{x})$ is referred to as the new "observables" of the data. The SVM algorithm then learns the hyperplanes that optimally split the data into distinct clusters in this new space. The hyperplane function in this case is then (Brunton 2019):

$$f(x) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b \tag{10}$$

This method of building nonlinear classifiers by enriching in higher-dimensions leads to a computationally intractable optimization (Brunton 2019). Thus computing the vectors $\mathbf{w}$ is prohibitively expensive and may not even be represented explicitly in memory. The "kernel trick" solves this problem by first representing $\mathbf{w}$ in terms of weighting parameters α_j that weight the different nonlinear observable functions $\Phi(x_j)$. Subsequently a kernel function can be defined that allows for one to operate in a high- dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between all pairs of data in the feature space (Brunton 2019). This results in a computationally tractable optimization.

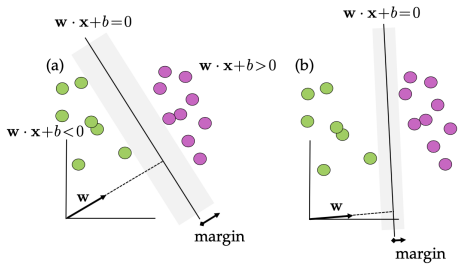

Figure 2: Two-class SVM discrimination task where a hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ is constructed that optimally separates the labeled data. The area of the margin separating the labeled data is maximal in (a) and much less in (b). Determining the vector $\mathbf{w}$ and parameter b is the goal of the SVM optimization. The vectors touching the edge of the gray regions of are termed the support vectors. (Brunton 2019)

2.2.5 Decision Tree

The decision tree is a hierarchical construct that looks for optimal ways to split the data in order to provide a robust classification and regression (Brunton 2019). Suppose that we have the following (Brunton 2019):

$$\text{data}\,\{x_j \in \mathbf{R}_n, j \in \mathbf{Z} := \{1, 2, \cdots, m\}\}$$
$$\text{labels}\,\{y_j \in \{\pm 1\}, j \in \mathbf{Z}' \subset \mathbf{Z}\}$$

The basic decision tree algorithm begins by scanning through each feature of the data for each data instance vector x_j in order to identify the value of x_j that best predicts the label y_j. Next, the prediction accuracy is compared for each split on the feature x_j and the feature giving the best segmentation of the data is selected as the split for the tree. Now with the two new branches of the tree created, this process is repeated on each branch. The algorithm terminates once each individual data point is a unique cluster, known as a leaf, on a new branch of the tree (Brunton 2019).

2.3 Classifier Performance Metrics

In assessing the performance of the classification algorithms confusion matrices and Receiver Operating Characteristic Curves will be used.

2.3.1 Confusion Matrix

A confusion matrix for a classification problem simply indicates the frequency of correct and erroneous class predictions in the test data. The counts indicate how often a class is predicted as itself or if it is frequently misclassified as another class. This is important information in assessing the performance and thus suitability of the classification model. In addition, Accuracy, Precision, Recall, and F1 metrics will be reported on. These metrics depend on the frequencies of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) classification cases. Taken together these give a more holistic view of classification performance.

$$\textbf{Accuracy: } \frac{TP + TN}{TP + TN + FP + FN}$$

$$\textbf{Precision: } \frac{TP}{TP + FP}$$

$$\textbf{Recall: } \frac{TP}{TP + FN}$$

$$\textbf{F1: } 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Accuracy alone does not offer insight into the number predictions that are falsely classified positive and falsely classified negative. As such precision and recall supplement the accuracy information by determining the correctly classified cases out of all predictions for a given class (precision) and the correctly classified cases out of all classification instances of a given class (recall). The F1 score gives more weight to false negatives and false positives while not letting large numbers of true negatives influence the score in an attempt to strike a balance between precision and recall. F1 is a particularly good metric when class imbalance is present in the data used for a classification task.

2.3.2 Receiver Operating Characteristics Curve

A Receiver Operating Characteristics (ROC) curve is a plot of False Positive Rate (FPR) versus True Positive Rate (TPR) for the classification of a given data class. These rates can be calculated using a macro-average strategy where the metric will be determined independently for each class before taking the average, or with a micro-average which instead collects all the contributions regardless of class before calculating the average FPR and TPR. In the case where a class imbalance may exist, the micro-average is the preferred method for calculating the ROC curve as it will not all the contributions of one class to disproportionately skew the curve trajectory. The ROC curve is a probability curve that indicates the relationship between the distributions of true positive and true of negative classifications. The area under the curve (AUC) is thus a measure of the degree of separability between a positive and negative classification.

2.4 Clustering Methods

2.4.1 K-Means

The goal of the k-means clustering algorithm is to take a set of vector valued data and group the observations into a specified number, k, clusters. In each iteration of the algorithm, each observation is labeled as belonging to a cluster with the nearest mean. In general the procedure of each iteration begins from a set of initial values for the mean of each cluster. From these values labels are assigned to each data observation based on their distances to each cluster mean. The new cluster centroids (means) are then calculated and used to reassign cluster labels for the next iteration. This process effectively minimizes that within cluster sum of

squares. The loss function can thus be written as (Brunton 2019):

$$\arg\min_{\mu_j} \sum_{j=1}^{k} \sum_{\mathbf{x}_j \in \mathcal{D}'} \|\mathbf{x}_j - \mu_j\|^2 \tag{11}$$

where the μ_j denote the mean of the j_{th} cluster and $\mathcal{D}'_j$ denotes the subdomain of data associated with that cluster.

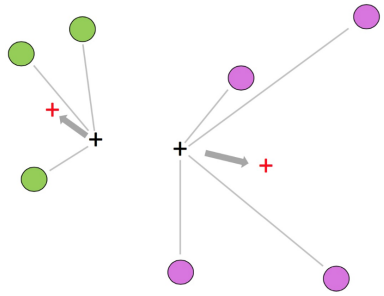

Figure 3: Example of the k-means algorithm for two clusters (k = 2). Starting values are indicated by black + markers. From these initial values each point in the space is determined to belong to either of the two clusters. After this labelling, the mean of the each cluster is computed (marked by red +). This process is repeated until the means converge (Brunton 2019).

2.4.2 Gaussian Mixture Model

The basic assumption of the Gaussian Mixture Model (GMM) method is that data observations $\mathbf{x}_j$ are a mixture of a set of k processes that combine to form the measurement. Like k-means, the fitted GMM model requires a specified number of mixtures k and the individual statistical properties of each mixture. The algorithm behind the GMM is the Expectation-Maximization (EM) algorithm which computes the maximum-likelihood parameters of the fit statistical models. The algorithm recursively estimates the best parameters possible from an initial guess proceeding much like the k-means algorithm in that initial guesses for the mean and variance are given for the assumed k-distributions. The algorithm then recursively updates the weights of the mixtures versus the parameters of each mixture until convergence is achieved. The fundamental assumption of the mixture model is that the probability density function (PDF) for observations of data x_j is a weighted linear sum of a set of unknown distributions:

$$f\left(\mathbf{x}_j, \Theta\right) = \sum_{p=1}^{k} \alpha_p f_p\left(\mathbf{x}_j, \Theta_p\right) \tag{12}$$

where $f(\cdot)$ is the measured PDF, $f_p(\cdot)$ is the PDF of the mixture j, and k is the total number of mixtures. Each of the PDFs $f_j(\cdot)$ is weighted by α_p (i.e. $(\alpha_1 + \alpha_2 + \cdots + \alpha_k = 1)$) and parametrized by an unknown vector of parameters Θ_p. To state the objective of mixture models more precisely then: Given the observed PDF $f(x_j, \Theta)$, estimate the mixture weights α_p and the parameters of the distribution Θ_p. One of the benefits of this modelling process is that the clustering ultimately provides distributions from which new data samples can be generated from. This is not immediately available following k-means clustering.

2.5 Clustering Algorithm Performance Metrics

In order to determine the most appropriate number of clusters to assume for the K-means algorithm the silhouette score will be computed for different assumed clusters and plotted to identify an optimal k value. For GMM the "elbow" method heuristic can be used to determine the optimum number of assumed mixtures from a plot of Akaike information criterion (AIC) and Bayesian information criterion (BIC) over different numbers of assumed mixtures. Once the optimal cluster number is determined for each algorithm, standard metrics

of performance are used to assess each model. These metrics are: homogeneity score, completeness score, v measure score, adjusted rand score, adjusted mutual info score, and silhouette score. The homogeneity metric of a cluster labeling given a ground truth determines how well a clustering result satisfies homogeneity. Homogeneity is satisfied by a result if all of the clusters contain only data points which are members of a single class. The completeness metric of a cluster labeling given a ground truth determines how well a clustering result satisfies completeness. A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster. The V-measure is the harmonic mean between the homogeneity and completeness score. The adjusted rand score is the Rand index adjusted for chance. The Rand Index computes a similarity measure between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings. The Adjusted Mutual Information (AMI) is an adjustment of the Mutual Information (MI) score to account for chance. It accounts for the fact that the MI is generally higher for two clusterings with a larger number of clusters, regardless of whether there is actually more information shared.

3 Algorithm Implementation and Development

3.1 SVD Face Data Analysis

The main steps of the algorithm are as follows:

1. Input the face image files by iterating through file directory, prepare for analysis
2. Flatten each image into single array, concatenate all images together
3. Compute SVD of flattened image data set, visualize top modes, estimate dimensionality
4. Project onto the left singular vectors, compare reconstructed images to originals

3.2 Classification Tasks

The main steps of the algorithm are as follows:

1. Input labels for classification (face id labels, gender labels)
2. Split data into training and testing sets
3. Implement K-Nearest Neighbors, Naive Bayes, LDA, Support Vector Machine, and Decision Tree classification methods
4. Investigate classifier performance using confusion matrix and ROC curves

4 Computational Results

4.1 SVD Face Data Analysis

The computational results in this section are a result of performing an SVD analysis of the cropped (Figure 4) and uncropped (Figure 5) Yale face images. The interpretation of the the left singular vectors of the U matrix are the principal reconstruction axes of the image data and the Σ values are the scalings of these axes. The right singular vectors of V apply rotation back into the original axes for the rank reduced reconstructions. The rank of the face space for the cropped Yale faces appears to be rank 4 based on Figure 4a. The rank of the uncropped face space appears to similarly be approximately 4 based on the top modes shown in Figure 5a.

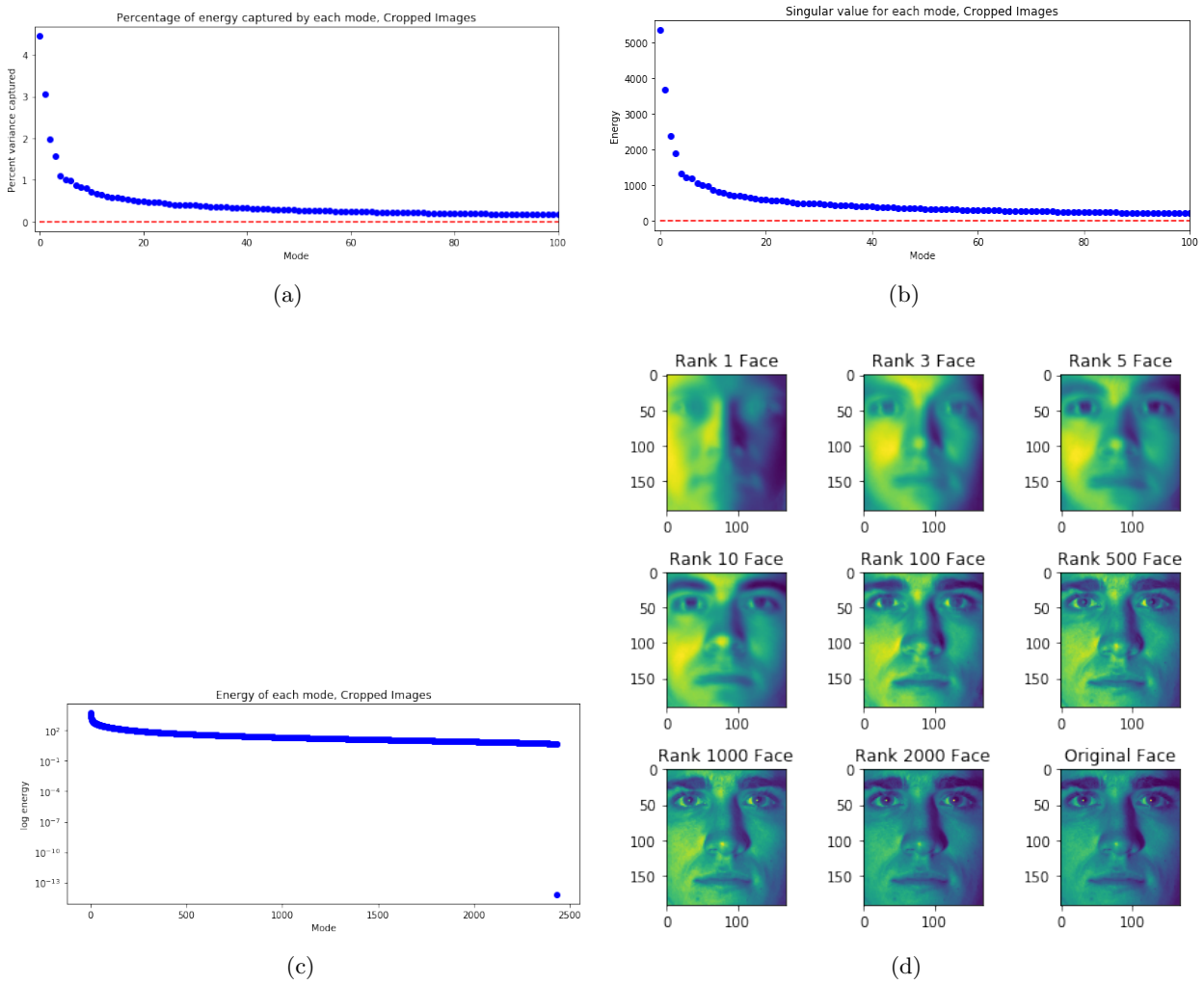

Figure 4: (a) The percentage of energy captured by each SVD mode. (b) Magnitudes of the singular values (c) Log energy of each SVD mode (d) Reconstructed faces with increasing ranks

4.2 Face Classification

4.2.1 (test 1) Face Identification

Classifiers were tested on rank 10, 50, and 100 reconstructions of the face image data set. The task for each classifier was to correctly associate each face image with the appropriate face identity. There were 38 different identities represented by the 2432 images. The best performing classifiers in each rank reconstruction case are illustrated in this section. The Support Vector Machine classifier was the best performer on rank 10 reconstructed data (Figure 6, Table 1). The Linear Discriminant Analysis Classifier was the best performer on rank 50 reconstructed data (Figure 7, Table 2). The Linear Discriminant Analysis Classifier also was the best performer on rank 100 reconstructed data (Figure 8, Table 3). Complete results for all classifiers tested are presented in Appendix B.

4.2.2 (test 2) Gender Classification

Classifiers were tested on rank 10, 50, and 100 reconstructions of the face image data set. The task for each classifier was to correctly associate each face image with the appropriate face gender (0: male, 1: female). This gender identity was not self reported by the participants and class labels were based on a subjective assessment prior to training the classifiers. This assessment does not intend to suggest that gender is a

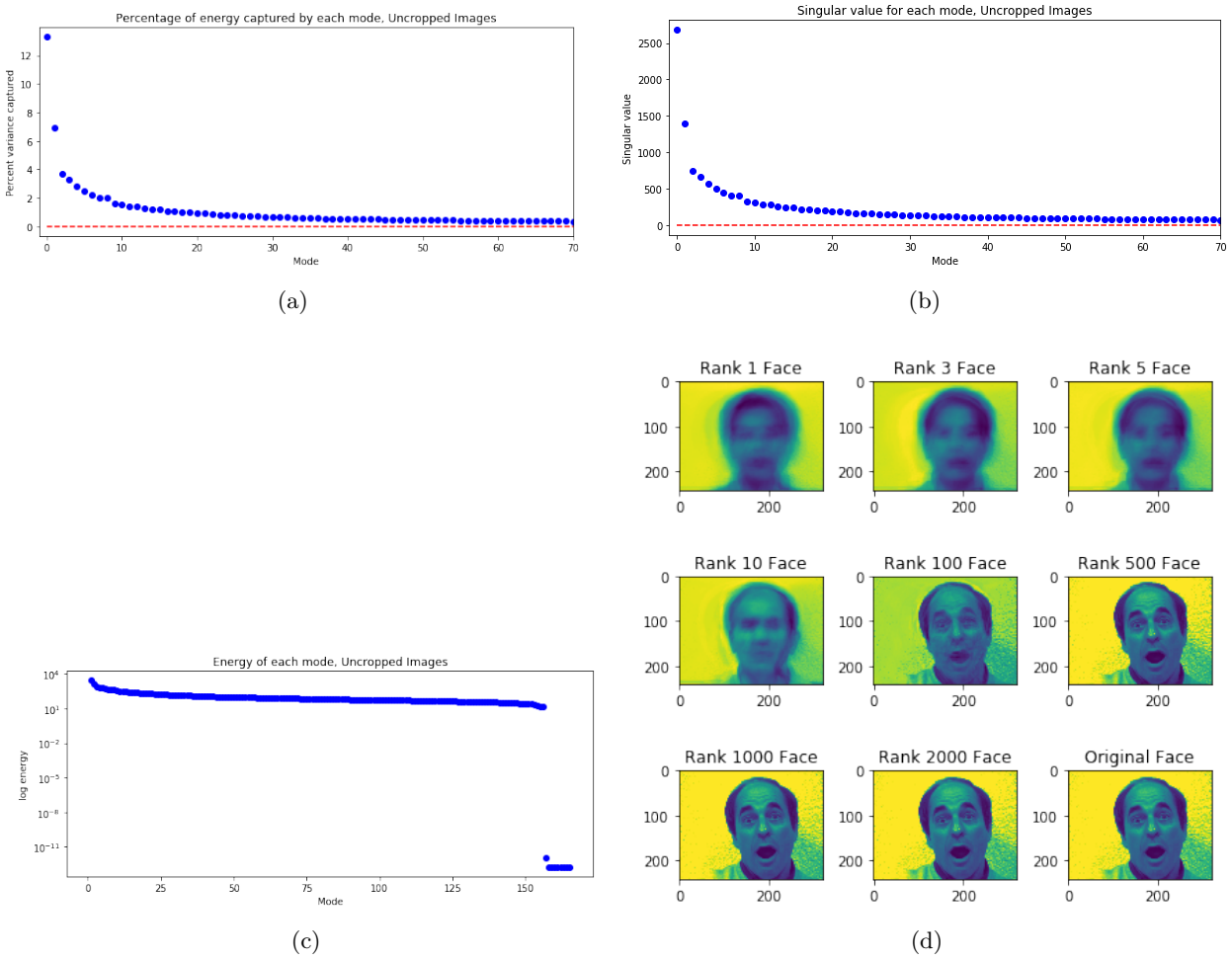

Figure 5: (a) The percentage of energy captured by each SVD mode. (b) Magnitudes of the singular values (c) Log energy of each SVD mode (d) Reconstructed faces with increasing ranks

binary label, nor that physical face appearance in any way must correlate with gender identity. Rather, this exercise seeks to use the crude example of binary gender as a simple two class classification case for illustration of the implementation of these machine learning algorithms. The best performing classifiers in each rank reconstruction case are illustrated in this section. The Decision Tree classifier was the best performer on rank 10 reconstructed data (Figure 9, Table 4). The Support Vector Machine classifier was the best performer on rank 50 reconstructed data (Figure 10, Table 5). The Support Vector Machine classifier also was the best performer on rank 100 reconstructed data (Figure 37, Table 6). Complete results for all classifiers tested are presented in Appendix B.

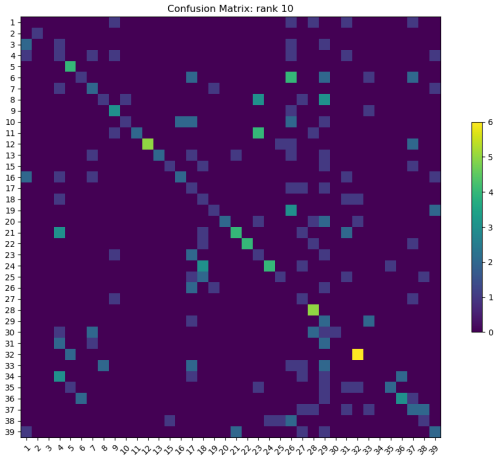

Figure 6: rank 10 Support Vector Machine Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.286885	0.313978	0.008697
Precision	0.449093	0.452264	0.030744
Recall	0.286885	0.313978	0.008697
F1	0.308009	0.318467	0.012042

Table 1: Table of rank 10 Support Vector Machine Classifier performance metrics for face identification.

4.2.3 (test 3) Unsupervised Algorithms

Two different unsupervised clustering algorithms were implemented: K-Means and Gaussian Mixture Models. Based on testing different cluster (k) values, $k = 5$ was determined to be the optimal cluster number for the K-Means clustering algorithm (Figure 12), and $k = 6$ was determined to be the optimal cluster number for the Gaussian Mixture Model algorithm (Figure 13). To visualize the clustering of each algorithm, both K-Means with 5 clusters and GMM with 6 clusters were applied to a rank 2 reconstruction of the face image data to illustrate the clustering in two dimensions (Figures **??**, 13). Performance metrics for these clustering algorithms on the rank 100 face image data set are delineated in Table 7 and Table 8 respectively.

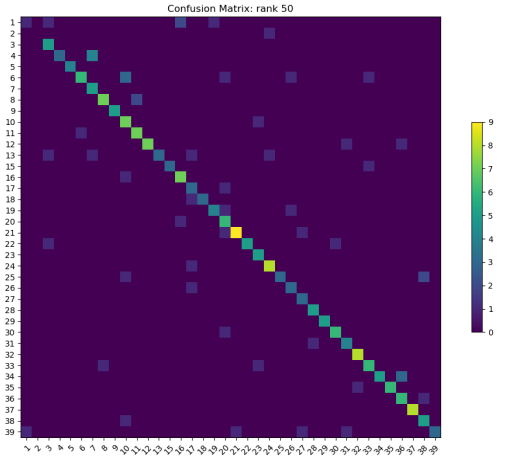

Figure 7: rank 50 Linear Discriminant Analysis Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.774590	0.774702	0.033563
Precision	0.820412	0.833561	0.023703
Recall	0.774590	0.774702	0.033563
F1	0.769527	0.780464	0.033446

Table 2: Table of rank 50 Linear Discriminant Analysis Classifier performance metrics for face identification.

5 Summary and Conclusions

5.1 SVD Face Data Analysis

Two significant results emerged from the face image SVD processing task. First, face image data can be adequately represented with rank 4 approximations. Second, the cropping of the face images is an important preprocessing/collection step for ensuring successful low rank image reconstructions. The misalignment of faces in each uncropped image contributed to significant differences in the low rank reconstructions (Figure 4d vs Figure 5d) generated through SVD. This is most assuredly due to increased noise in the data attributable to face orientation/position variability in each image frame.

5.2 Face Classification

Overall a face image data processing and algorithm implementation pipeline was successfully constructed capable of taking raw image files and classifying image instances from different rank reconstructions. All classification methods were largely successful with the most notable exception being the classifiers applied in the context of face identification with rank 10 approximated data. These were by far the poorest performing classifiers. The poor performance was assuredly exacerbated by the number of classes to predict (38) and the low dimensionality of data representation of the rank 10 data approximation used to train the classifiers. Also notable was the general poor performance of the Decision Tree classifiers on the face identification task. It is evident that Decision Tree classifiers struggle in instances where there are many class outputs to predict. This behavior might be modulated by optimizing the model parameters (e.g. max branch per node, max leafs per branch, etc.). In the siutation of binary gender classification the Decision Tree classifier performed the best on the lowest dimensionality data approximation highlighting the capability of the algorithm to classify data instances when fewer classes labels are present. In exploring the unsupervised algorithms, K-Means and GMM, different clusterings were observed. In general, the metrics illustrate poorly performing clustering algorithms. Specifically, the silhouette values for both K-Means and GMM were under 0.5 indicating that objects within a given cluster are not necessarily strikingly dissimilar to objects of other

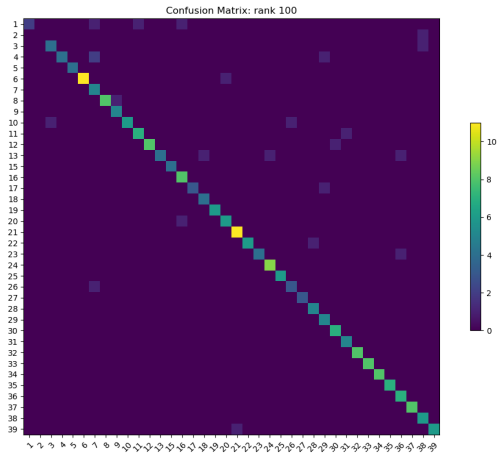

Figure 8: rank 100 Linear Discriminant Analysis Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.905738	0.905859	0.008627
Precision	0.920910	0.924838	0.008044
Recall	0.905738	0.905859	0.008627
F1	0.901648	0.907671	0.009169

Table 3: Table of rank 100 Linear Discriminant Analysis Classifier performance metrics for face identification.

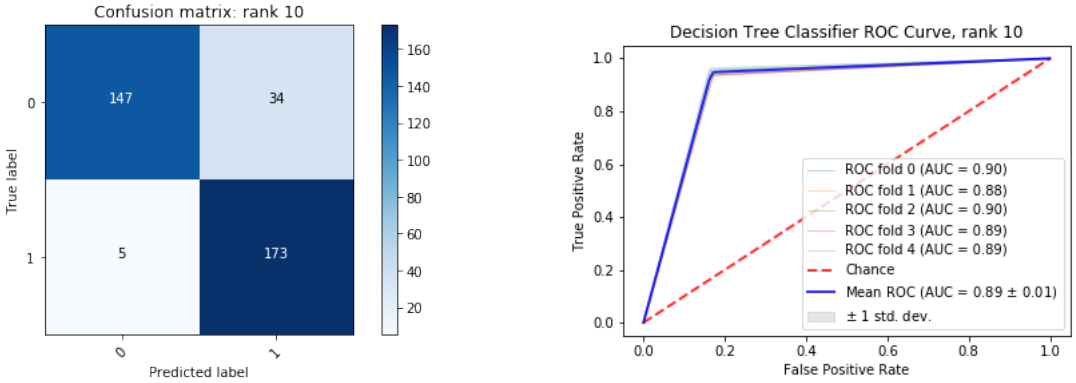

Figure 9: Test 2: (Left) Confusion matrix indicating predicted gender class counts for Decision Tree Classifier trained on a rank 10 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Decision Tree Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.891365	0.889302	0.009587
Precision	0.901976	0.894233	0.009186
Recall	0.891365	0.889302	0.009587
F1	0.890726	0.888963	0.009639

Table 4: Table of rank 10 Decision Tree Classifier performance metrics for gender classification.

clusters. The AIC/BIC plot for GMM models of increasing cluster sizes did not peak for any particular cluster value indicating that generally not improvement in modelling differences between data clusters was achieved. Similarly, a spike in the Silhouette score for the K-means algorithms at increase k values would be

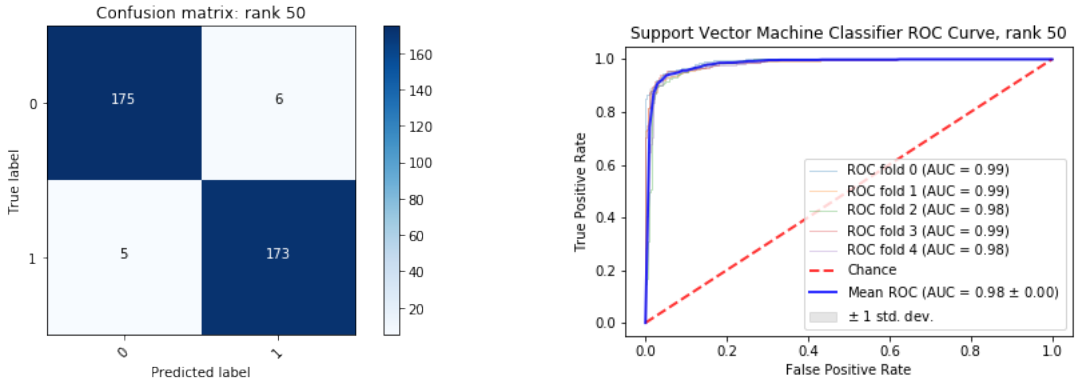

Figure 10: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Support Vector Machine Classifier trained on a rank 50 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Support Vector Machine Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.969359	0.940775	0.007030
Precision	0.969375	0.941215	0.006904
Recall	0.969359	0.940775	0.007030
F1	0.969360	0.940764	0.007042

Table 5: Table of rank 50 Support Vector Machine Classifier performance metrics for gender classification.

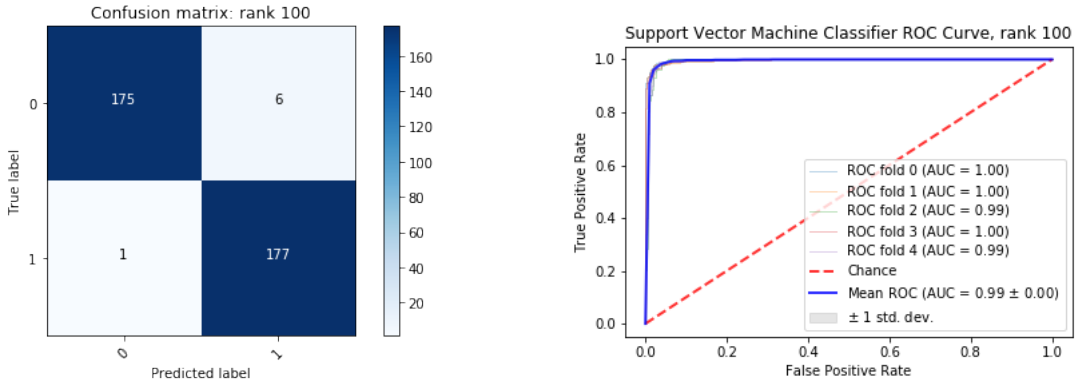

Figure 11: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Support Vector Machine Classifier trained on a rank 100 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Support Vector Machine Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.980501	0.971783	0.003721
Precision	0.980879	0.971991	0.003794
Recall	0.980501	0.971783	0.003721
F1	0.980500	0.971780	0.003724

Table 6: Table of rank 100 Support Vector Machine Classifier performance metrics for gender classification.

expected if an optimal cluster number was settled apon. In this assessment a rank 100 data approximation was used instead of the raw data which could have impacted the performance of these methods. Also, only k values from 2 to 15 were tested. It could be the case that the optimal cluster value is outside of this range,

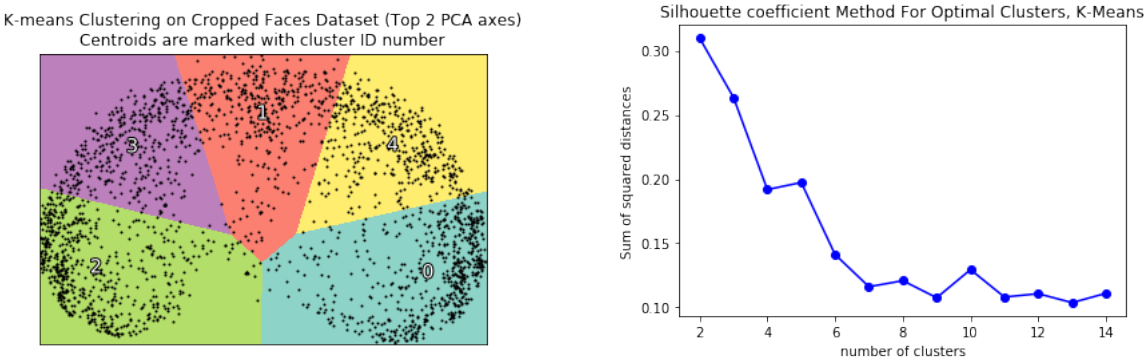

Figure 12: Test 3: (Left) Scatter plot of face image data along the first two principal components. The different colors indicate the clustering of the K-Means clustering algorithm on the rank 2 approximation of the face image data for k = 5. The number value identifiers indicate the cluster centroids, and the borders between colors indicate the decision boundary for the algorithm for each cluster. (Right) K-Means algorithms were run with increasing number of clusters to assess the optimum number of clusters. Silhouette scores are plotted for each cluster number with the optimum cluster number indicated by the subtle increase in score at k = 5.

Metric	Results
Homogeneity	0.001204
Completeness	0.000444
V-measure	0.000649
Adjusted Rand index (ARI)	0.001554
Adjusted Mutual Information (AMI)	-0.000122
Silhouette	0.189880

Table 7: Table of rank 100 K-Means Clustering metrics.

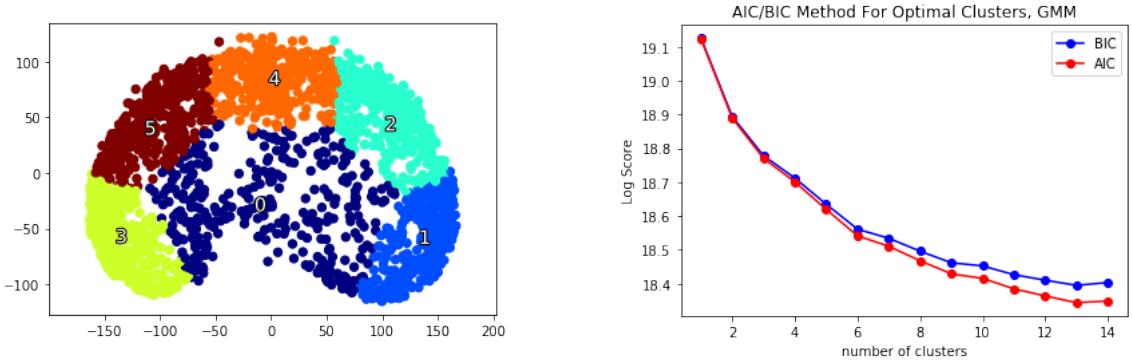

Figure 13: Test 3: (Left) Scatter plot of face image data along the first two principal components. The different colors indicate the clustering of the Gaussian Mixture Model clustering algorithm on the rank 2 approximation of the face image data for k = 6. The number value identifiers indicate the cluster centroids for each individual cluster. (Right) Gaussian Mixture Models were fit with increasing number of clusters to assess the optimum number of clusters. AIC and BIC scores are plotted for each cluster number with the optimum cluster number indicated by the "elbow" in the graph at k = 6.

though this is not likely based on the trend of the plots across the range tested.

Metric	Results
Homogeneity	0.004888
Completeness	0.001578
V-measure	0.002385
Adjusted Rand index (ARI)	0.002744
Adjusted Mutual Information (AMI)	0.001514
Silhouette	0.382625

Table 8: Table of rank 100 Gaussian Mixture Model Clustering metrics.

References

Brunton, Steven L. (Steven Lee) (2019). *Data-driven science and engineering : machine learning, dynamical systems, and control*. Cambridge, United Kingdom ; New York, NY: Cambridge University Press. ISBN: 9781108422093.

Kutz, Jose Nathan (2013). *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press.

Appendix A Python Functions

- `numpy.linalg.svd(a, full_matrices=True, compute_uv=True, hermitian=False)`
  Singular Value Decomposition.

- `sklearn.discriminant_analysis.LinearDiscriminantAnalysis(solver='svd', shrinkage=None, priors=None, n_components=None, store_covariance=False, tol=0.0001)`
  Linear Discriminant Analysis. A classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes' rule. The model fits a Gaussian density to each class, assuming that all classes share the same covariance matrix. The fitted model can also be used to reduce the dimensionality of the input by projecting it to the most discriminative directions.

- `sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)`
  Classifier implementing the k-nearest neighbors vote.

- `sklearn.naive_bayes.GaussianNB(*, priors=None, var_smoothing=1e-09)`
  Gaussian Naive Bayes (GaussianNB)

- `sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None)`
  C-Support Vector Classification.

- `sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort='deprecated', ccp_alpha=0.0)`
  A decision tree classifier.

- `sklearn.cluster.KMeans(n_clusters=8, *, init='k-means++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='deprecated', verbose=0, random_state=None, copy_x=True, n_jobs='deprecated', algorithm='auto')`
  K-Means clustering.

- `sklearn.mixture.GaussianMixture(n_components=1, *, covariance_type='full', tol=0.001, reg_covar=1e-0 max_iter=100, n_init=1, init_params='kmeans', weights_init=None, means_init=None, precisions_init=No random_state=None, warm_start=False, verbose=0, verbose_interval=10)`
  Gaussian Mixture.

- `sklearn.metrics.roc_auc_score(y_true, y_score, average='macro', sample_weight=None, max_fpr=None, multi_class='raise', labels=None)`
  Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores. This implementation can be used with binary, multiclass and multilabel classification, but some restrictions apply (see Parameters).

Appendix B Supplemental Figures and Tables

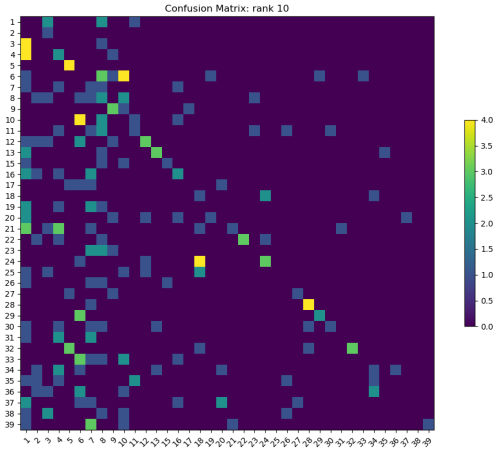

Figure 14: rank 10 K-Nearest Neighbors Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.176230	0.202449	0.02045
Precision	0.269004	0.313798	0.042107
Recall	0.176230	0.202449	0.020450
F1	0.186364	0.218037	0.022606

Table 9: Table of rank 10 K-Nearest Neighbors Classifier performance metrics for face identification.

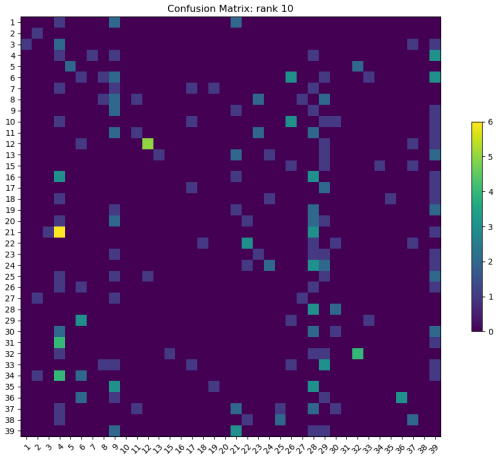

Figure 15: rank 10 Naive Bayes Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.139344	0.193304	0.021558
Precision	0.212186	0.308701	0.038860
Recall	0.139344	0.193304	0.021558
F1	0.146826	0.200973	0.023494

Table 10: Table of rank 10 Naive Bayes Classifier performance metrics for face identification.

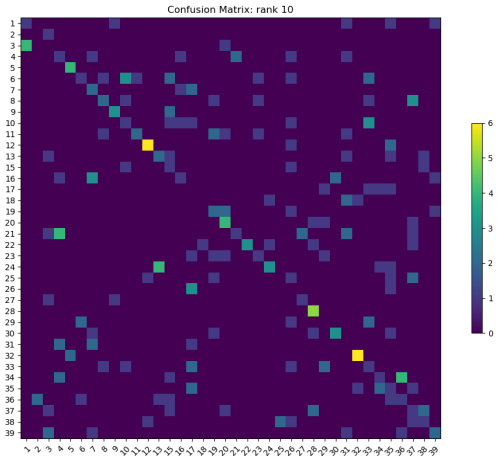

Figure 16: rank 10 Linear Discriminant Analysis Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.262295	0.338207	0.013366
Precision	0.314917	0.372363	0.017357
Recall	0.262295	0.338207	0.013366
F1	0.268377	0.335908	0.011737

Table 11: Table of rank 10 Linear Discriminant Analysis Classifier performance metrics for face identification.

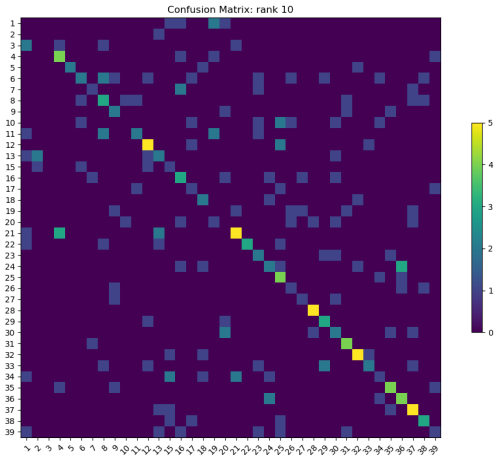

Figure 17: rank 10 Decision Tree Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.356557	0.383925	0.031742
Precision	0.365580	0.414585	0.027672
Recall	0.356557	0.383925	0.031742
F1	0.344385	0.385223	0.027790

Table 12: Table of rank 10 Decision Tree Classifier performance metrics for face identification.

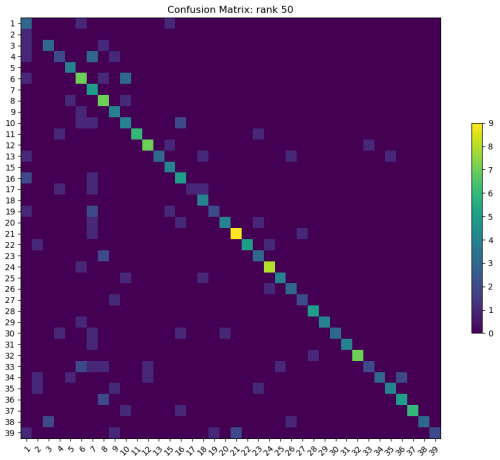

Figure 18: rank 50 K-Nearest Neighbors Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.643443	0.644437	0.020848
Precision	0.739660	0.735688	0.025636
Recall	0.643443	0.644437	0.020848
F1	0.649491	0.657720	0.019592

Table 13: Table of rank 50 K-Nearest Neighbors Classifier performance metrics for face identification.

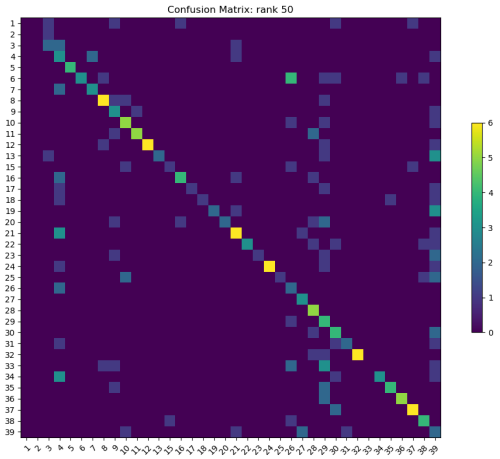

Figure 19: rank 50 Naive Bayes Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.491803	0.524707	0.027421
Precision	0.680075	0.663372	0.017899
Recall	0.491803	0.524707	0.027421
F1	0.510710	0.547730	0.025242

Table 14: Table of rank 50 Naive Bayes Classifier performance metrics for face identification.

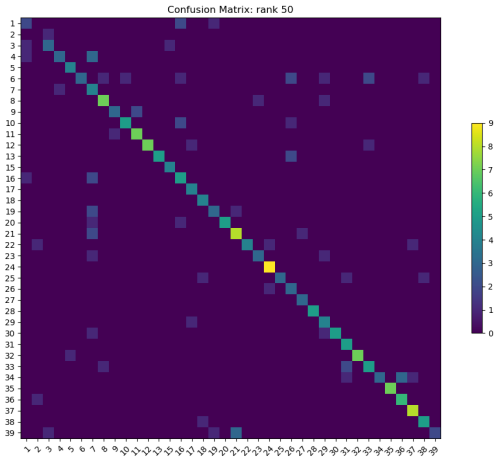

Figure 20: rank 50 Support Vector Machine Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.709016	0.755039	0.017495
Precision	0.786622	0.814328	0.016326
Recall	0.709016	0.755039	0.017495
F1	0.708370	0.759946	0.016177

Table 15: Table of rank 50 Support Vector Machine Classifier performance metrics for face identification.

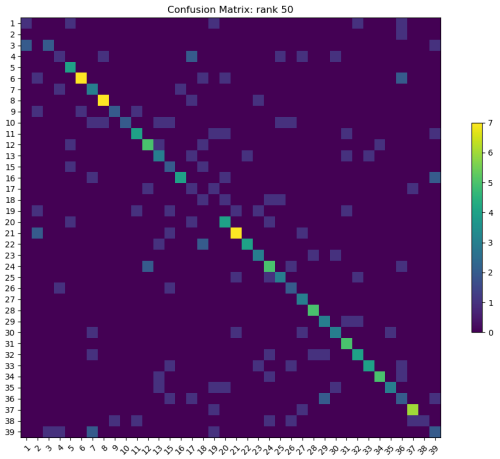

Figure 21: rank 50 Decision Tree Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.504098	0.522834	0.024622
Precision	0.574276	0.552282	0.0244856
Recall	0.504098	0.522834	0.024622
F1	0.508552	0.524648	0.022819

Table 16: Table of rank 50 Decision Tree Classifier performance metrics for face identification.

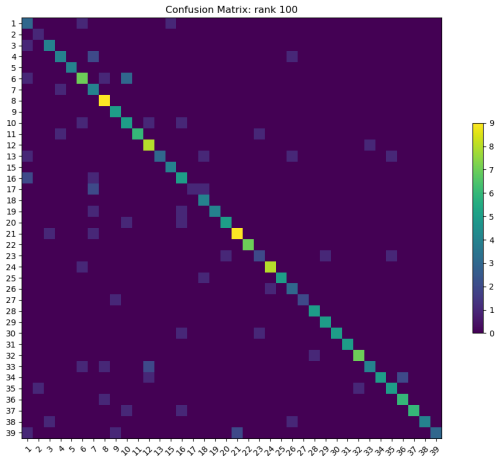

Figure 22: rank 100 K-Nearest Neighbors Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.745902	0.740414	0.015207
Precision	0.798135	0.802674	0.017548
Recall	0.745902	0.740414	0.015207
F1	0.747649	0.749133	0.015549

Table 17: Table of rank 100 K-Nearest Neighbors Classifier performance metrics for face identification.

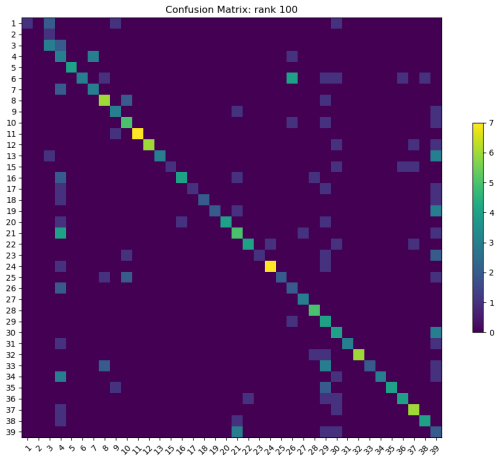

Figure 23: rank 100 Naive Bayes Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.540984	0.588231	0.025296
Precision	0.754781	0.711801	0.018674
Recall	0.540984	0.588231	0.025296
F1	0.574124	0.608171	0.021274

Table 18: Table of rank 100 Naive Bayes Classifier performance metrics for face identification.

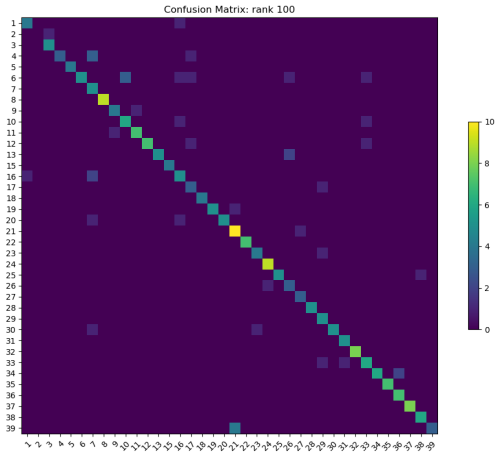

Figure 24: rank 100 Support Vector Machine Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.827869	0.840502	0.017278
Precision	0.868539	0.872261	0.014513
Recall	0.827869	0.840502	0.017278
F1	0.825824	0.841800	0.017390

Table 19: Table of rank 100 Support Vector Machine Classifier performance metrics for face identification.

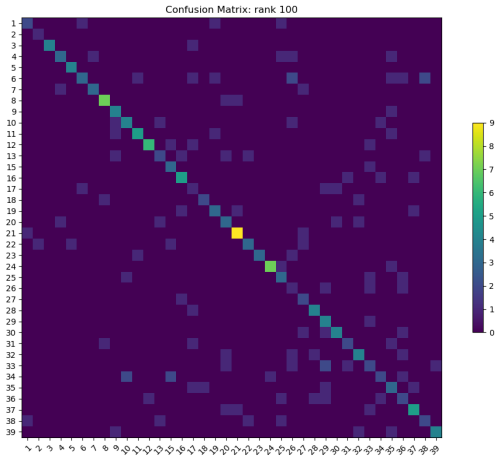

Figure 25: rank 100 Decision Tree Classifier Classifier confusion matrix for face identification.

Metric	Results	Average	Std
Accuracy	0.536885	0.531529	0.025553
Precision	0.574762	0.561601	0.019678
Recall	0.536885	0.531529	0.025553
F1	0.541536	0.531451	0.021723

Table 20: Table of rank 100 Decision Tree Classifier performance metrics for face identification.

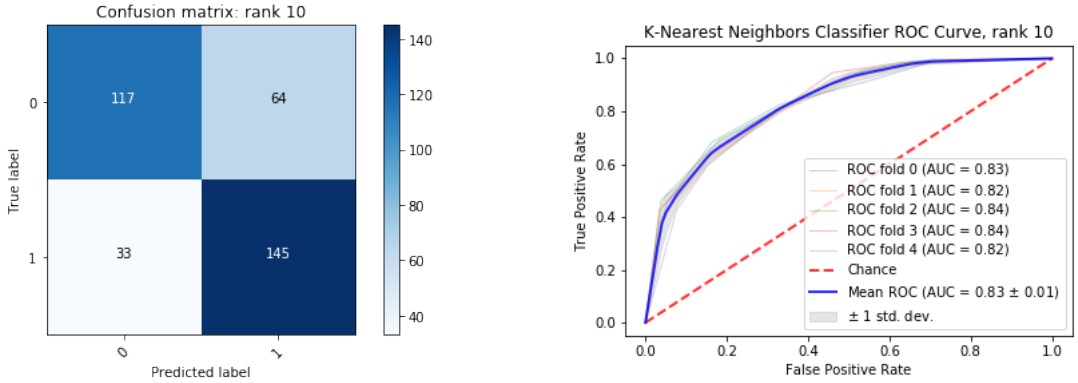

Figure 26: Test 2: (Left) Confusion matrix indicating predicted gender class counts for K-Nearest Neighbors Classifier trained on a rank 10 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for K-Nearest Neighbors Classifier, trained on the same rank 10 approximation data, across cross validation.

Metric	Results	Average	Std
Accuracy	0.729805	0.738605	0.003747
Precision	0.737250	0.745347	0.0068508
Recall	0.729805	0.738605	0.003747
F1	0.727974	0.736857	0.004458

Table 21: Table of rank 10 K-Nearest Neighbors Classifier performance metrics for gender classification.

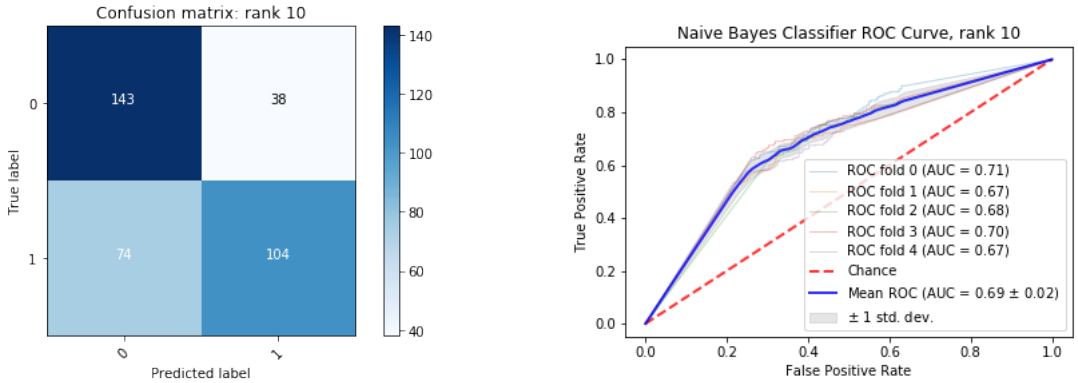

Figure 27: Test 2: (Left) Confusion matrix indicating predicted gender class counts for Naive Bayes Classifier trained on a rank 10 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Naive Bayes Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.688022	0.663876	0.013846
Precision	0.695384	0.667692	0.012356
Recall	0.688022	0.663876	0.013846
F1	0.684583	0.661917	0.015000

Table 22: Table of rank 10 Naive Bayes Classifier performance metrics for gender classification.

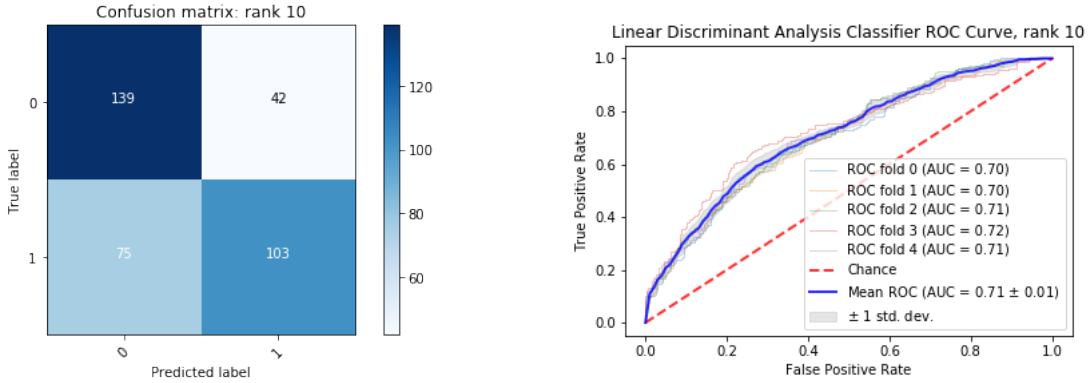

Figure 28: Test 2: (Left) Confusion matrix indicating predicted gender class counts for Linear Discriminant Analysis Classifier trained on a rank 10 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Linear Discriminant Analysis Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.674095	0.657054	0.019150
Precision	0.679685	0.661052	0.017997
Recall	0.674095	0.657054	0.019150
F1	0.671060	0.654883	0.020203

Table 23: Table of rank 10 Linear Discriminant Analysis Classifier performance metrics for gender classification.

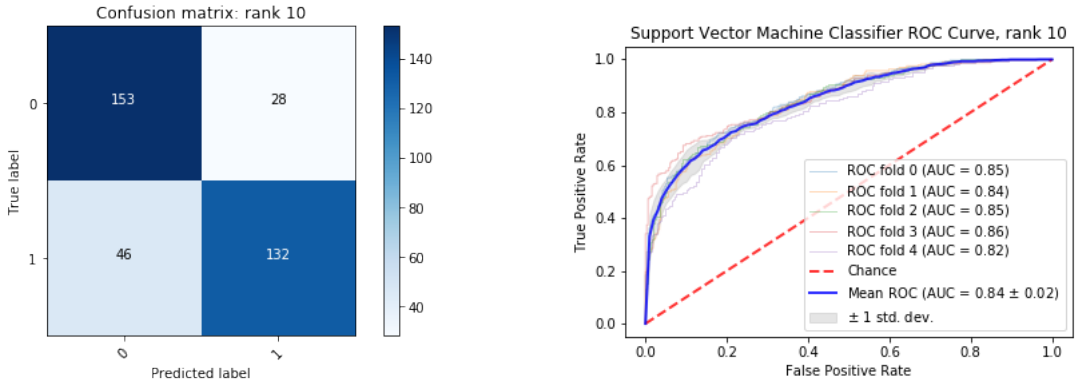

Figure 29: Test 2: (Left) Confusion matrix indicating predicted gender class counts for Support Vector Machine Classifier trained on a rank 10 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Support Vector Machine Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.793872	0.757209	0.016070
Precision	0.796687	0.763715	0.016141
Recall	0.793872	0.757209	0.016070
F1	0.793265	0.755761	0.016613

Table 24: Table of rank 10 Support Vector Machine Classifier performance metrics for gender classification.

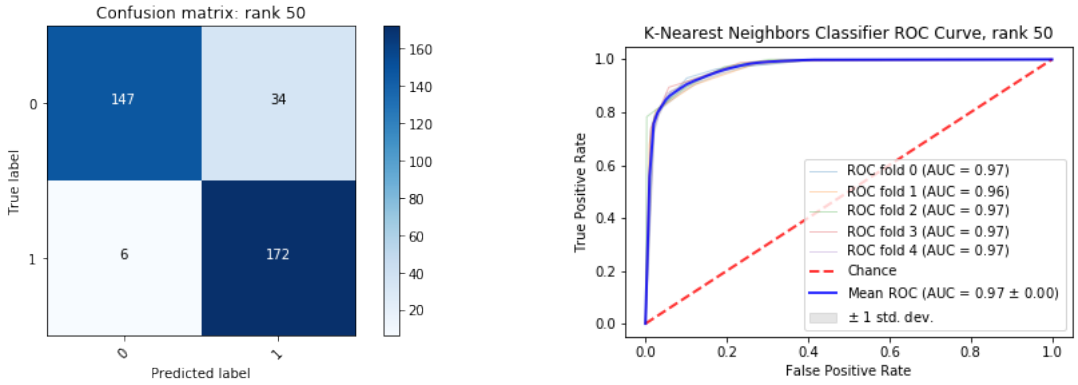

Figure 30: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the K-Nearest Neighbors Classifier trained on a rank 50 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same K-Nearest Neighbors Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.888579	0.893023	0.003252
Precision	0.898394	0.898035	0.0036046
Recall	0.888579	0.893023	0.003252
F1	0.887971	0.892710	0.003383

Table 25: Table of rank 50 K-Nearest Neighbors Classifier performance metrics for gender classification.

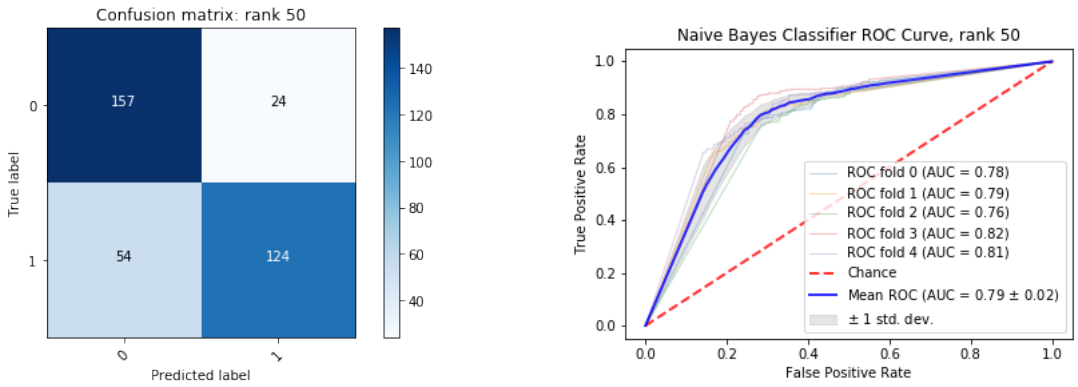

Figure 31: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Naive Bayes Classifier trained on a rank 50 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Naive Bayes Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.782730	0.750078	0.017359
Precision	0.790565	0.754240	0.017052
Recall	0.782730	0.750078	0.017359
F1	0.781047	0.749054	0.0176573

Table 26: Table of rank 50 Naive Bayes Classifier performance metrics for gender classification.

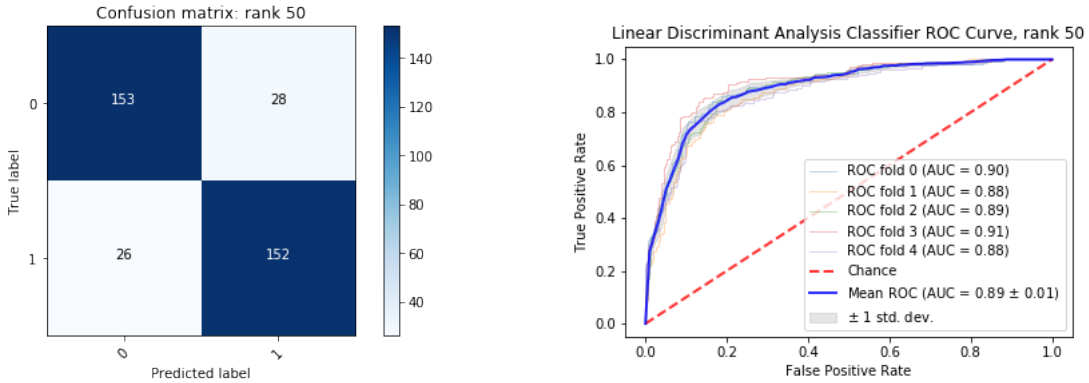

Figure 32: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Linear Discriminant Analysis Classifier trained on a rank 50 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Linear Discriminant Analysis Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.849582	0.826357	0.015962
Precision	0.849640	0.826930	0.015642
Recall	0.849582	0.826357	0.015962
F1	0.849585	0.826286	0.0159863

Table 27: Table of rank 50 Linear Discriminant Analysis Classifier performance metrics for gender classification.

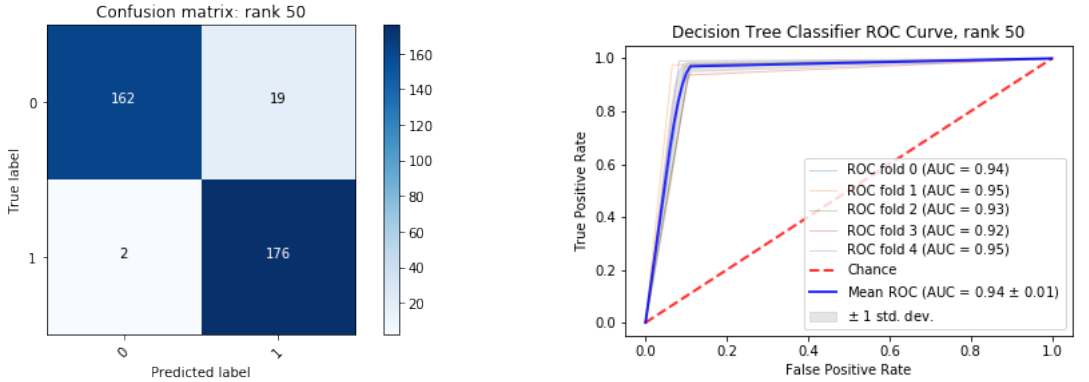

Figure 33: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Decision Tree Classifier trained on a rank 50 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Decision Tree Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.941504	0.935814	0.010124
Precision	0.945541	0.938513	0.010397
Recall	0.941504	0.935814	0.010124
F1	0.941396	0.935708	0.010176

Table 28: Table of rank 50 Decision Tree Classifier performance metrics for gender classification.

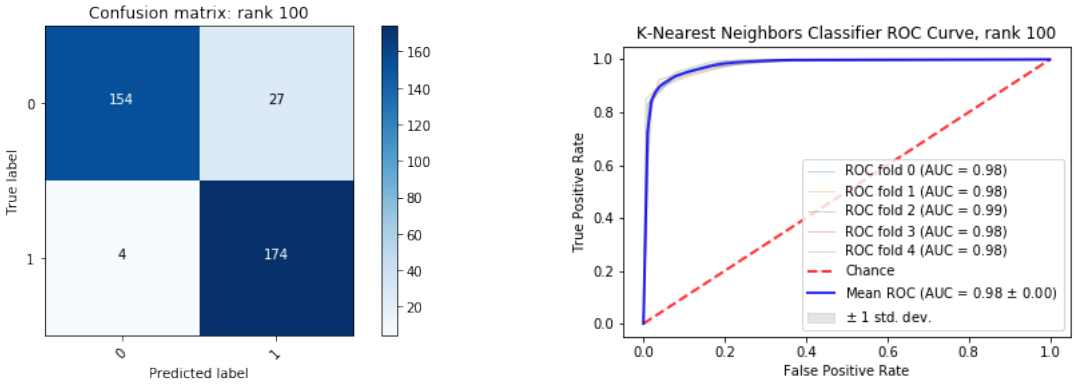

Figure 34: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the K-Nearest Neighbors Classifier trained on a rank 100 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same K-Nearest Neighbors Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.913649	0.914419	0.009271
Precision	0.920633	0.919146	0.007227
Recall	0.913649	0.914419	0.009271
F1	0.913340	0.914192	0.0093406

Table 29: Table of rank 100 K-Nearest Neighbors Classifier performance metrics for gender classification.

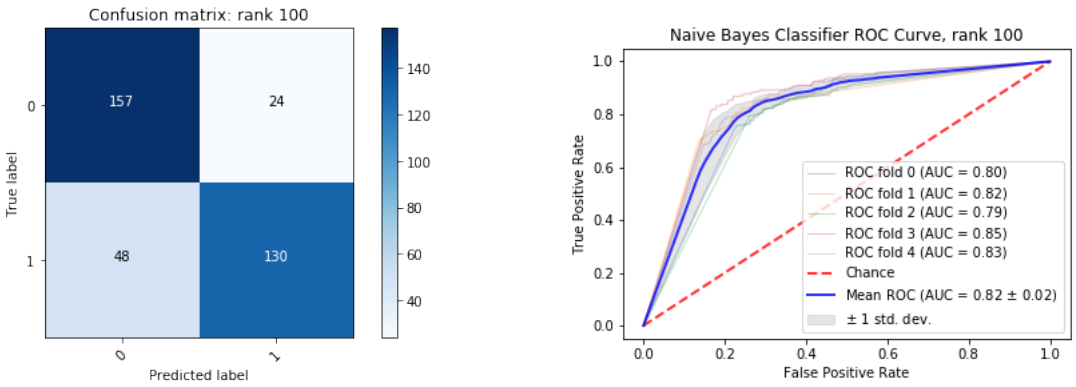

Figure 35: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Naive Bayes Classifier trained on a rank 100 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Naive Bayes Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.799443	0.779225	0.022820
Precision	0.804678	0.782622	0.022004
Recall	0.799443	0.779225	0.022820
F1	0.798429	0.778497	0.023146

Table 30: Table of rank 100 Naive Bayes Classifier performance metrics for gender classification.

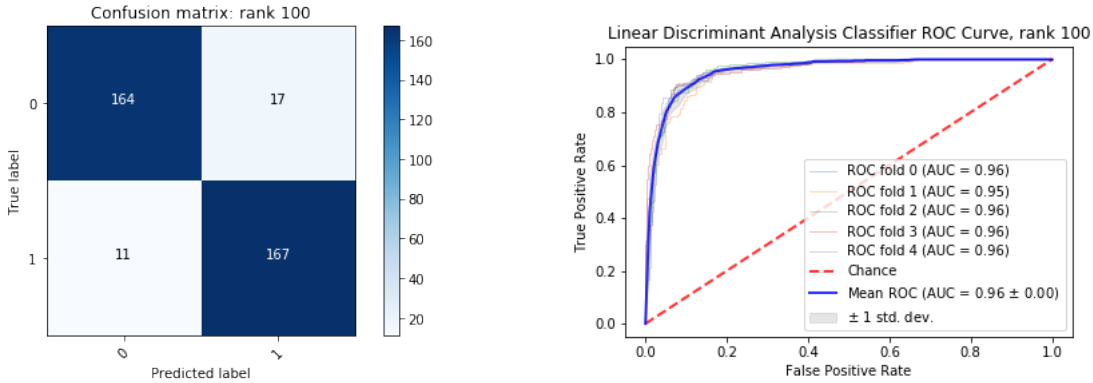

Figure 36: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Linear Discriminant Analysis Classifier trained on a rank 100 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Linear Discriminant Analysis Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.922006	0.897984	0.012065
Precision	0.922499	0.898462	0.012208
Recall	0.922006	0.897984	0.012065
F1	0.921995	0.897970	0.012060

Table 31: Table of rank 100 Linear Discriminant Analysis Classifier performance metrics for gender classification.

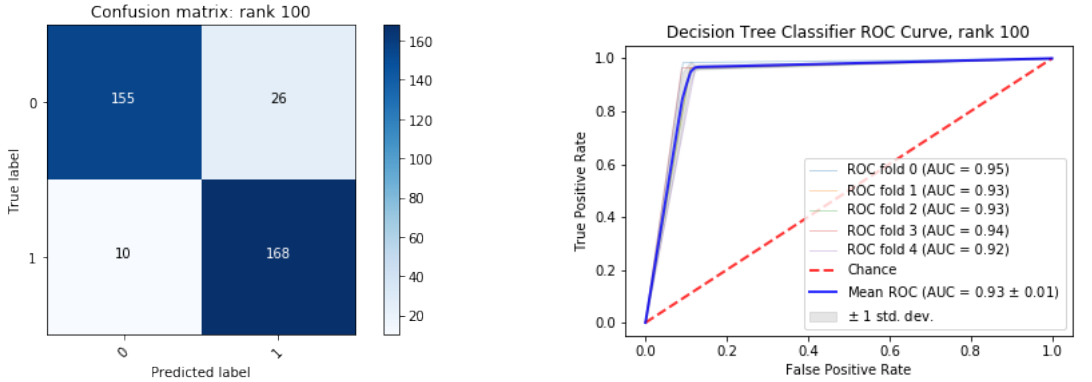

Figure 37: Test 2: (Left) Confusion matrix indicating predicted gender class counts for the Decision Tree Classifier trained on a rank 100 approximation of the face image data set. (Right) Receiver Operating Characteristic (ROC) curves for the same Decision Tree Classifier across cross validation.

Metric	Results	Average	Std
Accuracy	0.899721	0.936124	0.005751
Precision	0.902993	0.937861	0.006353
Recall	0.899721	0.936124	0.0057511
F1	0.899559	0.936062	0.005783

Table 32: Table of rank 100 Decision Tree Classifier performance metrics for gender classification.