Exploration of Dimension Reduction

Jingmei Yang
Systems Engineering
Boston University
jmyang@bu.edu

H M Sabbir Ahmad
Systems Engineering
Boston University
sabbir92@bu.edu

Hee Jae Kim
Electrical and Computer Engineering
Boston University
hjkim37@bu.edu

1. Introduction

Amongst various emerging technologies, image processing has been revolutionized with the advancement of machine learning and is widely utilized in interdisciplinary domains to automatically identify different types of signals. For example, medical imaging is an emerging application that can be revolutionized using machine learning by facilitating early diagnosis of disease from images based on the underlying patterns that are not easy to detect by a human being. In addition, autonomous cars utilize computer vision to navigate safely in a dynamic real-world environment by classifying various environmental contexts from constantly changing image streams.

However, conventional machine learning-based classification algorithms can perform poorly on high-dimensional data due to the problem of the curse of dimensionality. To circumvent the issue, we aim to employ feature selection in classification tasks to encapsulate useful information from original data into low-dimensional feature space. The compressed meaningful features are then used as input to off-the-shelf classification algorithms to classify images.

Towards this end, we explored two classes of dimensionality reduction algorithms namely: i. Unsupervised and ii. Supervised algorithms. We began with the conventionally used linear dimensionality reduction algorithm, PCA. Following that we explored nonlinear unsupervised algorithms as the underlying relationship between features can be nonlinear. We explored two categories of nonlinear dimensionality reduction algorithms which are i. Kernel PCA and ii. manifold based dimensionality reduction algorithm including MDS [1], Isomap [2], LLE [4]. Following that, we explored various supervised feature selection algorithms which can be categorized into three groups namely: i. Embedded feature selection, ii. Filter type feature selection and iii. Sequential feature selection. Embedded feature selection involves feature selection as part of the model training process and hence eliminates the necessity for training an additional classifier for the low dimensional data. On the other hand, filter type feature selection computes feature importance to generate the low dimensional embedding which is used to train a classifier. Finally, sequential algorithms recursively add/prune the dataset into the desired dimensionality.

We explored the performance of classification algorithms combined with various feature selection techniques in the medical imaging domain. In particular, we adopted a widely used breast cancer classification dataset and sampled the images to construct a balanced binary classification dataset. For the comparison, we generated two datasets of different sizes where each of them containing a total of 6,000 and 20,000 images respectively.

The report is composed of six sections. In the following section, we present the problem tackled along with details of various dimensionality reduction algorithms implemented as part of the project. In the subsequent section, we include the implementation details which is followed by the experimental results section where we include the details of the dataset used for the project along with the results obtained using the various explored algorithms. The key finding of the work is finally summarized in the conclusion section.

2. Problem Formulation and Solution Approaches

Our primary objective was to explore various dimensionality reduction techniques for large datasets. Towards this end, we mainly investigated two different classes of algorithms namely i. Unsupervised and ii. Supervised dimensionality reduction algorithms. The secondary aim was to analyze how these algorithms affect the performance of a classification problem applied to Medical Imaging. Provided in fig. 1 an illustration of the classification pipeline incorporated with dimensionality reduction. This section will describe the various algorithms we implemented for the two classes of approaches.

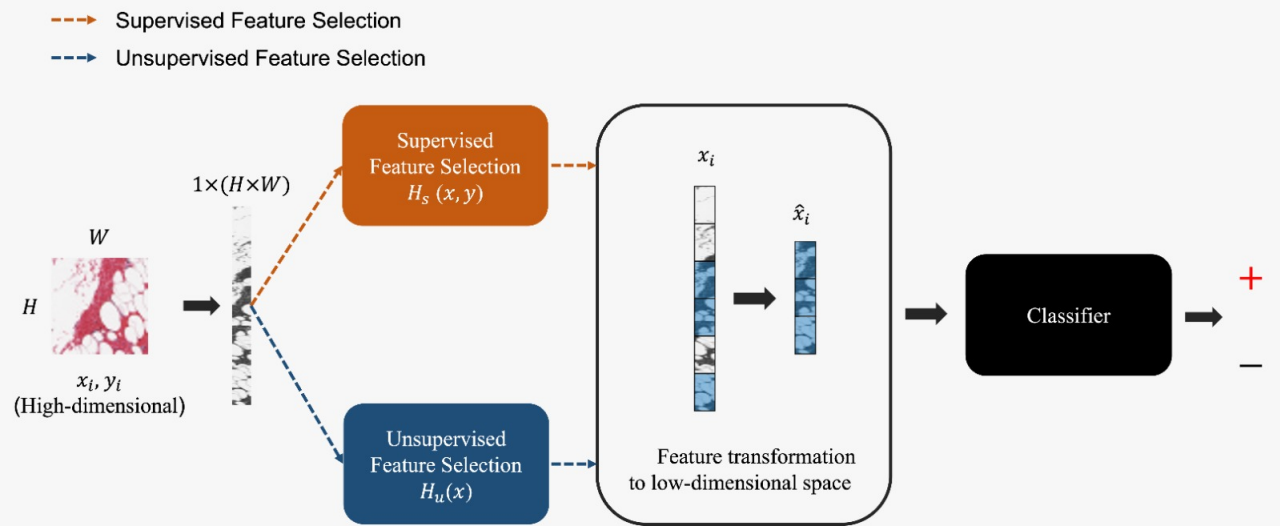

Figure 1: Illustration of dimensionality reduction with classification pipeline.

2.1. Baseline classifier: Logistic Regression

As part of our implementation we used logistic regression as our baseline classifier whose decision rule is given as follows:

$$P(y = 1|\boldsymbol{x}; \theta) = \frac{1}{1 + exp(\theta^T \boldsymbol{x})} \tag{1}$$

Learning problem: Given a set of feature vectors with labels:

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$$

the learning problem is to find the best θ that minimizes the negative log-likelihood of the training set given by:

$$l(\theta) = -\frac{1}{n} \sum_{i=1}^{n} (y_i \log (h_\theta (\boldsymbol{x_i})) + (1 - y_i) \log (1 - h_\theta (\boldsymbol{x_i}))$$

Where, $h_\theta (\boldsymbol{x_i}) = \frac{1}{1+\exp(\theta^T x_i)}$

2.2. Unsupervised Dimensionality Reduction Methods

The methods we explored under this class can be further divided into two subclasses namely i. linear and ii. nonlinear dimensionality reduction algorithms. Amongst linear approaches, we mainly explored Principle Component Analysis (PCA) for dimensionality reduction. The nonlinear algorithms we explored can be fundamentally categorized into the kernel and manifold-based methods which will be described in the remainder of this subsection.

2.2.1 KernelPCA

Kernel-PCA is an extension of PCA, which incorporates kernel methods for dimensionality reduction of a dataset that is not linearly separable. The fundamental idea behind Kernel-PCA is to use basis functions that raise the original dataset to a high dimensional space where it is linearly separable. However, the additional dimensionality also introduces computational overload as it introduces a higher number of algorithm parameters; and the original dataset is required to be mapped to the high dimensional space using basis functions. These constraints are overcome using kernel trick which involves computing the kernel function $K(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ such that: i. $K(\mathbf{u}, \mathbf{v}) = \langle \phi(\mathbf{u}), \phi(\mathbf{v}) \rangle$ for some $\phi(\cdot)$ and inner product $\langle \cdot\cdot \rangle$ and ii. the computational complexity of evaluating $K(\mathbf{u}, \mathbf{v}) = O(d) =$ complexity of evaluating inner product $\langle \mathbf{u}, \mathbf{v} \rangle$ the function $K(:)$ is called a (qualified) kernel function. As part of our implementation, we used the RBF kernel given as follows:

$$K_{\exp}(\mathbf{u}, \mathbf{v}) = \exp \left\{ -\gamma \|\mathbf{u} - \mathbf{v}\|_2^2 \right\}, \gamma > 0 \tag{2}$$

where $\mu, v \in R^n$. The value of γ was chosen to be 10 as part of our implementation. Kernel-PCA involves computing the nonnzero k eigenvalues $\lambda_1 \geq \cdots \geq \lambda_k > 0$ of $\frac{1}{n} C_n^{\mathrm{T}} \mathbb{K} C_n$ and corresponding orthonormal eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{R}^n$. The low dimensional space can be constructed by picking the first $m \leq k$ principal components.

Then, any test point $\mathbf{x}_{\text{test}} \in \mathbb{R}^d$ can be projected into

low dimensional space $\mathbb{R}^m$:

$$\hat{\mathbf{z}} = \begin{pmatrix} \frac{1}{\sqrt{n\lambda_1}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\sqrt{n\lambda_m}} \end{pmatrix} \begin{pmatrix} - & \mathbf{v}_1^{\mathrm{T}} & - \\ \vdots & \vdots & \vdots \\ - & \mathbf{v}_m^{\mathrm{T}} & - \end{pmatrix} C_n^{\mathrm{T}} \left(\mathbb{K}_{\mathbf{x}_{\text{test}}} - \frac{1}{m} \mathbb{K} \mathbf{1}_n \right)$$

where the ij^{th} element of $n \times n$ kernel feature matrix $\mathbb{K}$ is equal to $K(\mathbf{x}_i, \mathbf{x}_j)$ and - the i^{th} element of $n \times 1$ vector $\mathbb{K}_{\mathbf{x}_{\text{test}}}$ is equal to $K(\mathbf{x}_i, \mathbf{x}_{\text{test}})$

2.2.2 Multi-dimensional Scaling

Assuming we are given pairwise distances between any two points in a very high dimensional space, let's say, $\mathbf{x_1}, \cdots, \mathbf{x_n} \in \mathbb{R}^d$, we want to represent these two data points in a low dimensional space, such that:

$$\|\mathbf{z_i} - \mathbf{z_j}\| \approx d_{ij}$$

where d_{ij} is the distance between $\mathbf{x_i}$ and $\mathbf{x_j}$ in the original space, $\mathbf{z_i}$ and $\mathbf{z_j}$ are the corresponding two points in the embedding space. We want to find low-dimensional representations $\mathbf{z_i}, \cdots, \mathbf{z_j} \in \mathbb{R}^k$, for some $k < d$, such that the distance between $\mathbf{z_i}$ and $\mathbf{z_j}$ in the embedding space is approximately equal to the distance between $\mathbf{x_i}$ and $\mathbf{x_j}$ in the original space.

Let $\mathbf{Z} = [\mathbf{z_1}, \cdots, \mathbf{z_N}]^T$ be the embedding matrix. The classical MDS works as follows:

$$\mathbf{G} = \mathbf{Z}\mathbf{Z}^T = -\frac{1}{2}\mathbf{C}_n\mathbf{D}\mathbf{C}_n \tag{3}$$

where $\mathbf{C} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, and $\mathbf{D}$ is the squared pairwise distance matrix in the original space. Since $\mathbf{G}$ needs to be positive definite in order to have a solution, $\mathbf{G}$ can be rewritten as:

$$\mathbf{G} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}\Lambda^{\frac{1}{2}}\mathbf{U}^T \tag{4}$$

If $k > r$ = rank(G), the MDS has a exact solution:

$$\mathbf{Z} = \mathbf{U}_k\mathbf{\Lambda}_k^{\frac{1}{2}} = [\sqrt{\lambda_1}\mathbf{u_1}, \cdots, \sqrt{\lambda_r}\mathbf{u_r}, \mathbf{0}, \cdots, \mathbf{0}]$$

where λ_i is the i_{th} largest eigenvalue, and u_i is the corresponding eigenvector.

Note that if we want to preserve the pairwise Euclidean distances of the original data, the MDS is equivalent to PCA.

2.2.3 ISOMAP

ISOMAP is just a special case of MDS. The idea behind ISOMAP is very similar to the idea behind MDS. By default, we use Euclidean distance in MDS, we want the pairwise Euclidean distance between any two points in the original space to be approximately equal to the distance in the embedding space. In ISOMPA, we try to preserve the geodesic distance rather than the Euclidean distance, as the Euclidean distance between these two points is not the true distance. The geodesic distance captures the true, non-linear geometry corresponding to the curved dimension, as the geodesic distance measures the shortest distance between two points along the manifold. Often time, it is hard to find the exact geodesic distances since we may not know the true manifold. Instead, we form a graph based on the nearest neighbors and then find the shortest distance between any two points by applying the shortest path algorithm, like Dijkstra's algorithm. After we get the distance matrix for all pairs of data points, we apply MDS to the distance matrix.

Algorithm 1 ISOMAP

Input: Pairwise distances $d_X(i,j)$ of data points in the input space, embedding dimension $k \geq 1$, neighborhood graph method (kNN)

Output: k-dimensional representation of the data $\mathbf{Z} \in \mathbb{R}^{n \times k}$

Step:

1. Construct a neighborhood graph G from the given distances $d_X(i,j)$.
2. Compute the shortest-path distances $d_G(i,j)$ between all vertices of G by using Dijkstra's algorithm.
3. Apply MDS with $d_G(i,j)$ as input distances to find a k-dimensional representation $\mathbf{Z}$ of the original data

2.3. Supervised Feature Selection/ Dimensionality Reduction

Fundamentally, the algorithms we explored can be categorized into three groups. The first group of algorithms we looked at are embedded feature selection algorithms. These algorithms learn feature importance as part of the model learning process. This is achieved by training and obtaining the importance of the features in the trained model. This type of algorithm selects features that work well with a particular learning process. The second group of algorithms comes under the filter type feature section, and as the name suggests these algorithms select/filter out features based on their importance computed using certain characteristics of the features, such as feature variance and feature relevance to the response. However, unlike embedded feature selection algorithms, the features are selected as part of a data preprocessing step, and then a model is trained using the selected features. Therefore, filter type feature selection is uncorrelated to the training algorithm. The final group of algorithms we looked at performs feature selec-

tion/dimensionality reduction sequentially. This group of algorithms requires a classifier that is used to compute coefficients/importance values for the actual feature set. The algorithm starts training using a subset of features and then adds or removes a feature using a selection criterion based on the coefficients/importance values. The training process is repeated until the stopping criteria are satisfied. The remainder of this subsection will describe the specific algorithms we explored under the three aforementioned techniques.

2.3.1 Embedded feature selection

a. LASSO Regression: This approach involves augmenting L-1 regularization term to the cost function of the linear regression model to select the most important features in the dataset. Thus the learning problem becomes the following:

Learning problem: Given a set of feature vectors with labels:

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\} \text{ where } \boldsymbol{x_i} \in \mathbb{R}^d \text{ and } y_i \in \{0, 1\}$$

find a sparse θ using L-1 norm regularization. Hence, the cost function becomes the following:

$$l(\theta) = \lambda \left(\sum_{\ell=1}^{d} |\boldsymbol{\theta}_\ell| \right) - \frac{1}{n} \sum_{i=1}^{n} \left(y_i \log \left(h_\theta \left(\boldsymbol{x_i} \right) \right) + (1 - y_i) \log \left(1 - h_\theta \left(\boldsymbol{x_i} \right) \right) \right)$$

Where, $h_\theta \left(\boldsymbol{x_i} \right) = \frac{1}{1+\exp(\theta^T x_i)}$
The decision rule is

$$h_\theta \left(\boldsymbol{x}_{\text{test}} \right) = P(y = 1 \mid \boldsymbol{x}; \theta) = \frac{1}{1 + \exp \left(\theta^T \boldsymbol{x}_{\text{test}} \right)} \tag{5}$$

b. Linear Discriminant Analysis: This method is similar to PCA except LDA finds directions that maximize the Signal-to-Noise ratio (SNR). For a dataset with K-classes the K means μ_k are vectors in $\mathcal{R}^d$, and they lie in an affine subspace H of dimension at most $K - 1$.

Learning Problem: Given a set of feature vectors with labels:
$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ where $\boldsymbol{x_i} \in \mathbb{R}^d$, $y_i \in \mathcal{Y}$ and $\mathcal{Y} = \{1, \ldots, m\}$ find the $K - 1$ LDA directions

$$W_{LDA} = \begin{bmatrix} 1 & \cdots & | \\ \mathbf{w}_{LDA,1} & \cdots & \mathbf{w}_{LDA,m} \\ \mathrm{I} & \cdots & \mathrm{I} \end{bmatrix}$$

The d-dimensional data is then reduced to p-dimensions where $p \leq K - 1$ as following:

$$\tilde{\boldsymbol{x}} = W_{LDA,\text{ reduced}}^{\mathrm{T}} \cdot \boldsymbol{x} \tag{6}$$

Where,

$$W_{LDA,\text{ reduced}} = \begin{bmatrix} | & \cdots & | \\ \mathbf{w}_{LDA,1} & \cdots & \mathbf{w}_{LDA,p} \\ | & \cdots & | \end{bmatrix}$$

The LDA direction for the binary dataset is given as:

$$\mathbf{w}_{LDA} = \left(\widehat{\mathrm{S}}_{x.\text{avg}} \right)^{-1} (\widehat{\boldsymbol{\mu}}_{x2} - \widehat{\boldsymbol{\mu}}_{x1})$$

Where, $\widehat{\boldsymbol{\mu}}_{x1}$ is the empirical $d \times 1$ class-1 mean vector, $\hat{\boldsymbol{\mu}}_{x2}$ is the empirical $d \times 1$ class- 2 mean vector, $\hat{S}_{x1}$ is the empirical $d \times d$ class-1 covariance matrix and $\hat{S}_{x2}$ is the empirical $d \times d$ class-2 covariance matrix.

2.3.2 Filter type feature selection

a. Neighborhood Component Analysis: Neighborhood Component Analysis (NCA) is a supervised feature selection algorithm that projects the high dimensional data to low dimensional space using a linear transformation that optimizes a criterion related to the leave-one-out accuracy of the nearest neighbor classifier on a training set. Therefore the learning problem is given as follows:

Learning Problem: Given a labeled dataset $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ where $\boldsymbol{x_i} \in \mathbb{R}^d$, $y_i \in \{0, 1\}$ find the matrix A such that:

$$\tilde{\boldsymbol{x}} = A\boldsymbol{x},\ \tilde{\boldsymbol{x}} \in \mathbb{R}^p,\ p \leq d \tag{7}$$

This projection matrix A defines a Mahalanobis distance metric that can be used by the nearest neighbor classifier in the projected space [5]. NCA selects a single neighbor stochastically and the probability a point i selects another point j as its neighbor among k is given by:

$$p_{ij} = \frac{\exp \left(- \left\| Ax_i - Ax_j \right\|^2 \right)}{\sum_{\mathrm{k} \neq \mathrm{j}} \exp \left(- \left\| Ax_i - \mathrm{A}x x_{\mathrm{k}} \right\| \right)} \tag{8}$$

The stochastic selection rule aids in finding the probability p_i that a point i will be correctly classified [3].

$$p_i = \sum_{j \in Y_i} p_{ij} \text{ where } y_i = \{j \mid y_i = y_j\} \tag{9}$$

The objective of NCA is to maximize the expected number of points correctly classified. The objective (cost) function is given by,

$$f(A) = \sum_i \sum_{j \in Y_i} p_{ij} = \sum_i p_i \tag{10}$$

2.3.3 Sequential feature selection

a. Recursive Feature Elimination The algorithm begins with the full set of features and trains the classifier and obtains the importance of each feature. Then, the least important features are pruned from the current set of features. This procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached. The general steps in the algorithm have been outlining in algorithm 2

Algorithm 2 Recursive Feature Elimination

Input: $\mathcal{D}_{\text{normalized}} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$, $z :=$ number of features to be selected, classifier $h_\theta(\boldsymbol{x_i})$ parameterized by θ

Output: $X_{\text{reduced}} = [\tilde{\boldsymbol{x}}_\mathbf{1}\ \tilde{\boldsymbol{x}}_\mathbf{2}\ \ldots\ \tilde{\boldsymbol{x}}_\boldsymbol{d}]$ where $\tilde{x}_i \in \mathbb{R}^z$

Variables: Variables: $z_{iter} :=$ number of features in current iteration, $l(\theta) :=$ loss function for the classifier. $X_{iter} = [\boldsymbol{x}_\mathbf{1}\ \boldsymbol{x}_\mathbf{2}\ \ldots\ \boldsymbol{x}_\boldsymbol{d}] :=$ dataset after every iteration.

Steps:
While $z_{\text{iter}} \geq z$:
i. $\boldsymbol{\theta}_{\text{iter}} = \operatorname{argmin} l(\boldsymbol{\theta})$: fit $\boldsymbol{\theta}$ using the classifier $\boldsymbol{h_\theta}(\boldsymbol{x_i})$ on the dataset.
ii. $f : \theta_{i,\text{ iter}} \rightarrow \pi_{i,\text{ iter}}\, \forall i \in [1, z_{iter}]$: map classifier coefficients to importance values.
iii. $X_{iter} = \left[\begin{array}{cccc} x_1 & x_2 & \ldots & x_{z_{iter}} \end{array}\right] \backslash x_i$ where $i = \operatorname{argmin}(\pi_{i,iter})$: remove feature i corresponding to lowest importance value.
iv. $z_{iter} = z_{iter} - 1$: decrement the feature count by 1

Return $X_{\text{reduced}} = X_{\text{iter}}$;

3. Implementation

The images are of dimension 50x50 pixels which are flattened first and converted into 2500 dimensional vectors each. In the next stage, the flattened high-dimensional image vectors are embedded into low-dimensional space by either supervised or unsupervised dimensionality reduction algorithms. The dimensionality of the embedding varies depending on the chosen algorithm. Additionally, depending on the chosen algorithm, a classifier is trained. For instance, embedded-type feature selection doesn't require any additional classifier training following the feature selection process. We split both datasets where 60% of the data is used for training while the remaining data is evenly split and used for cross-validation and testing.

4. Experimental Results

4.1. Dataset

In this project, we utilized invasive ductal carcinoma (IDC) dataset, which is widely used as the most common subtype of all breast cancers. The original dataset consists of 62 whole-mount slide images of breast cancer and 277,524 patches of size 50×50 were extracted. Among the entire patches, 198,738 patches and 78,786 patches are labeled as IDC negative and IDC positive, respectively. To focus on our project goal in investigating various feature selection algorithms, we selected the same amount of patches from each class to generate balanced data sets. We especially constructed two datasets of different sizes and compared their performance; the dataset denoted as IDC3000 and IDC10000 contain 3,000 and 10,000 images per each class, respectively.

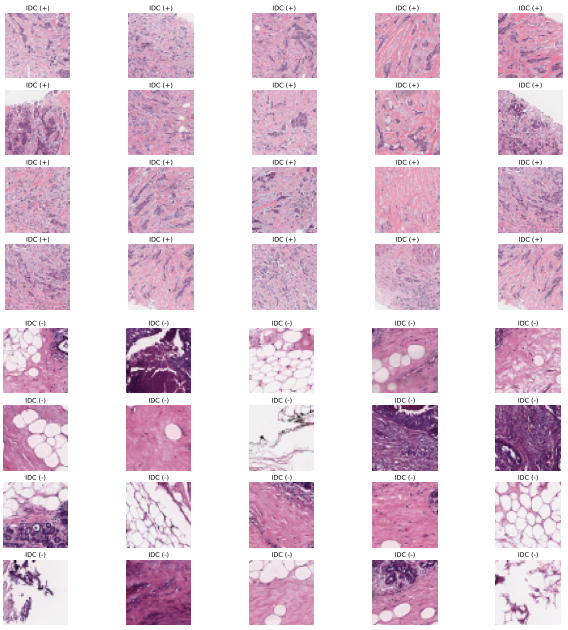

Figure 2: Image visualization

4.2. Performance Metric

We used the correct classification rate (CCR) as our performance metric to evaluate the performances of different dimension reduction methods.

4.3. Comparisons on Unsupervised Dimension Reduction Methods

In this section, we discuss the differences in the performance of unsupervised dimension reduction methods from two angles: CCR and computational complexity.

Firstly, we compare the CCRs of different dimension reduction methods using logistic regression as the base-

line classifier. As demonstrated in Figure 3 and Figure 4, as the number of key components increases, CCR increases. We reasoned that as the dimension increases, more information is kept and logistic regression has better discriminative power. Generally speaking, CCR is positively related to the number of components. In addition, kernel-PCA outperforms PCA. PCA is known to recover the true structure of data lying on or near a linear subspace of the high-dimensional space. In our dataset, the images are too complex to be captured well in a low-dimensional space in a linear fashion. However, non-linear methods, such as MDS and PCA with RBF kernel preserve more information. Interestingly, among all unsupervised dimension reduction methods, logistic regression with MDS achieves the best discriminative power, while ISOMAP captures the least information in the dataset.

Secondly, we compare the CCRs of different dimension reduction methods using a decision tree as the baseline classifier. As demonstrated in Figure 5 and Figure 6, KernelPCA and MDS combined with the decision tree achieve high CCRs. In contrast, the CCRs of ISOMAP and LLE combined with the decision tree fluctuate from 60% to 72%. Similarly, KernalPCA with the decision tree outperforms PCA with the decision tree.

Finally, we compare the computational complexity among unsupervised methods. As we apply different dimension reduction methods to the input images and then feed the transformed features as input into a classifier, running the whole pipeline to get a final classification result is computationally intensive. What's more, ISOMAP and LLE are neighbors-based methods, and the number of neighbors we pick for these two methods has a great influence on the performance. We ran cross-validation to tune this hyperparameter. It turns out 9 neighbors and 2 neighbors work best for ISOMAP and LLE, respectively. As shown in Figure 7, ISOMAP is the most computationally expensive method while other methods are relatively cheaper to transform the features from a high-dimension space to a low-dimension space.

4.4. Comparisons on Supervised Dimension Reduction Methods

In this section, we investigate the performance differences in supervised methods combined with logistic regression. As presented in Figure 8 and Figure 9, the pattern of LDA is similar to that of LASSO, and both methods are superior to other supervised methods, as LDA and LASSO are capable of capturing more underlying information in the dataset. More specifically, LDA selects the best projection direction that maximizes the signal-to-noise ratio, and LASSO also automatically picks the best projection direction. In addition, it can be clearly seen that the CCRs of logistic regression models

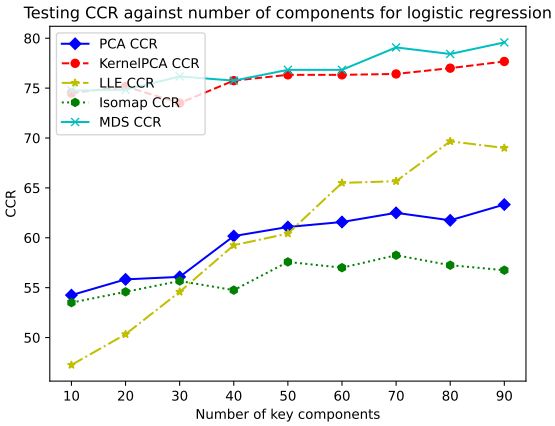

Figure 3: Testing CCR results of unsupervised methods on IDC3000 when using logistic regression as the baseline classifier

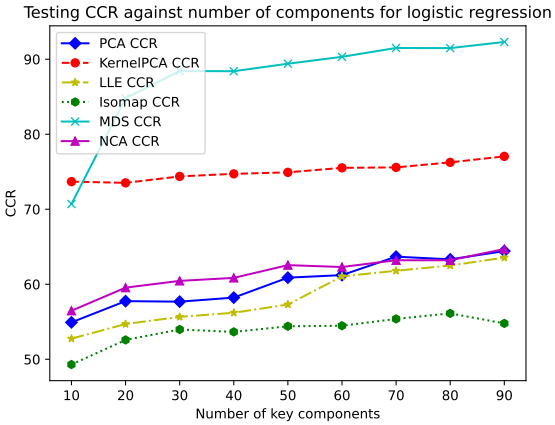

Figure 4: Testing CCR results of unsupervised methods on IDC10000 when using logistic regression as the baseline classifier

with NCA and with RFE keep increasing as the number of dimensions increases. We reasoned that as more dimensionalities are used in the transformation, more information is encapsulated in transformed features.

4.5. Comparisons on All Dimension Reduction Methods

When using logistic regression as a baseline classifier, LDA and LASSO are similar, and they retain the most important information to discriminate healthy images from cancerous images among all dimension reduction methods. When it comes to the decision tree, LDA is the best dimension reduction method, as it achieves the highest CCR when combined with the decision tree.

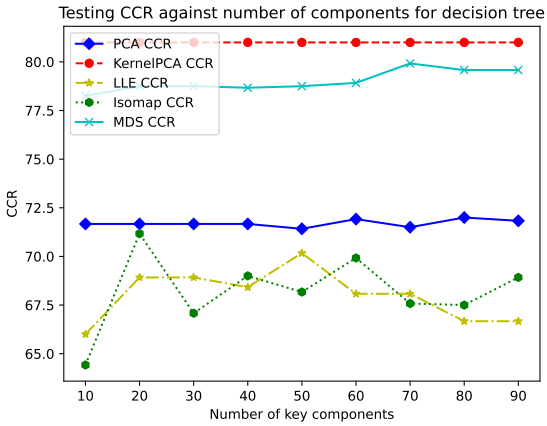

Figure 5: Testing CCR results of unsupervised methods on IDC3000 when using decision tree as the baseline classifier

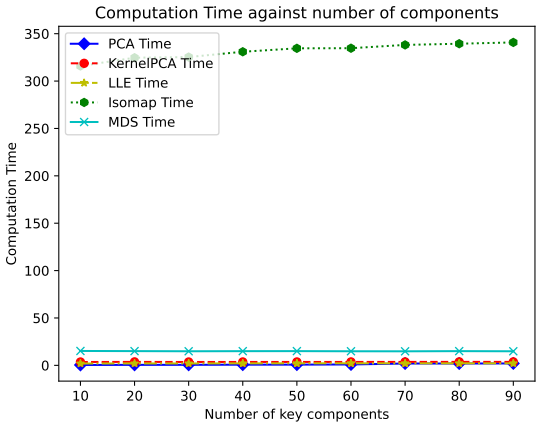

Figure 7: Comparisons on the computational complexity of unsupervised dimension reduction methods.

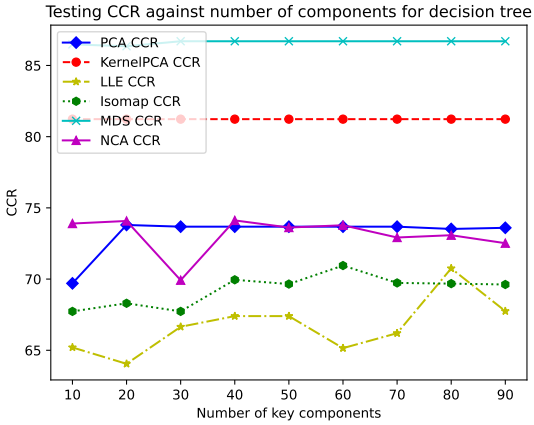

Figure 6: Testing CCR results of unsupervised methods on IDC10000 when using decision tree as the baseline classifier

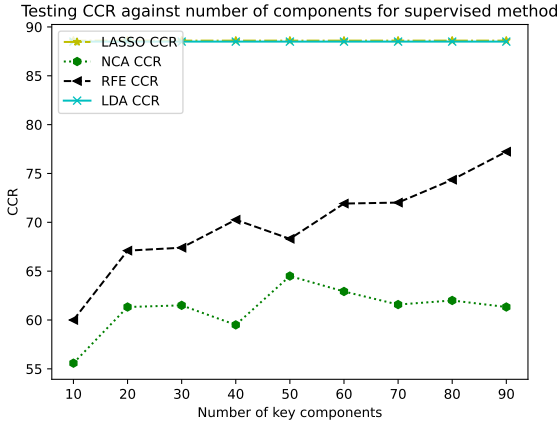

Figure 8: Testing CCR results of supervised methods on IDC3000 when using logistic regression as the baseline classifier

5. Conclusion

In this project, we explored how various feature selection techniques affect the performance of classification algorithms by transforming high-dimensional data into low-dimensional feature space with selected components. Amongst all feature selection techniques, the highest CCR was obtained using LASSO which is a supervised feature selection technique. The supervised feature selection methods achieve higher performance in terms of CCR compared to the unsupervised techniques as they utilize the best projection direction for data transformation. The results also demonstrated that most of the methods have an increasing trend in CCR as the number of selected components increases as more meaningful information will be encapsulated into the transformed data. Furthermore, we observed that the decision tree outperforms logistic regression as it takes a non-linear decision boundary for classification.

References

[1] F. Anowar, S. Sadaoui, and B. Selim. Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). *Computer Science Review*, 40:100378, 2021.

[2] M. Balasubramanian and E. L. Schwartz. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

[3] J. Goldberger, G. E. Hinton, S. Roweis, and R. R. Salakhutdinov. Neighbourhood components analysis. *Ad-*

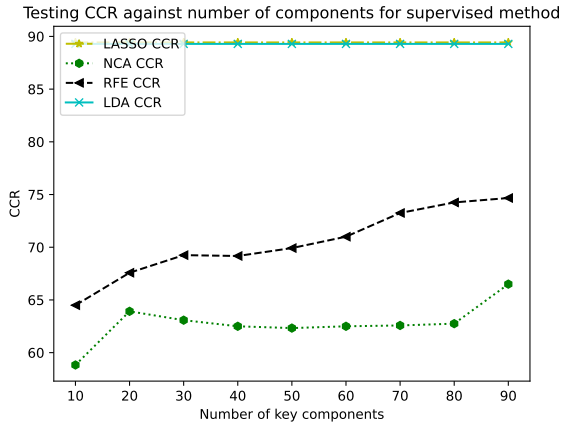

Figure 9: Testing CCR results of supervised methods on IDC10000 when using logistic regression as the baseline classifier

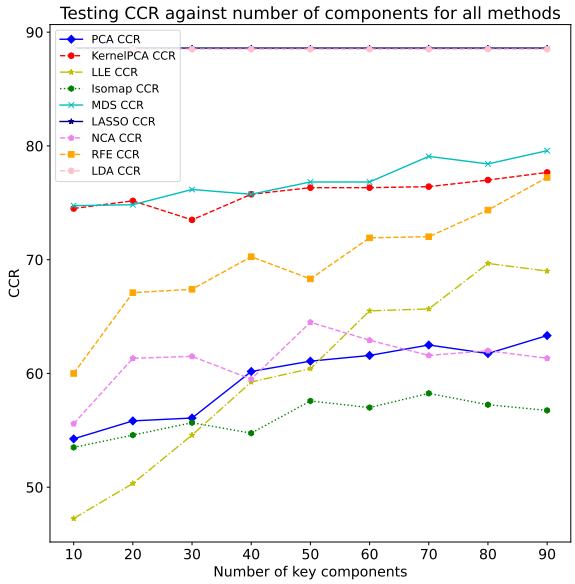

Figure 10: Testing CCR results of all methods on IDC3000 when using logistic regression as the baseline classifier

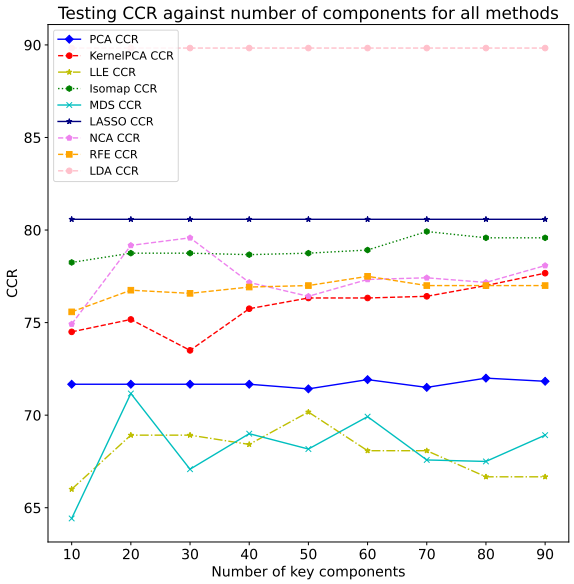

Figure 11: Testing CCR results of all methods on IDC3000 when using decision tree as the baseline classifier

vances in neural information processing systems, 17, 2004.

[4] L. K. Saul and S. T. Roweis. An introduction to locally linear embedding. *unpublished. Available at: http://www. cs. toronto. edu/~ roweis/lle/publications. html*, 2000.

[5] N. Singh-Miller, M. Collins, and T. J. Hazen. Dimensionality reduction for speech recognition using neighborhood components analysis. In *Eighth Annual Conference of the International Speech Communication Association*, 2007.