A comparative study for using PCA, LDA, GDA, and Lasso for dimensionality reduction before classification algorithms

Duyemo Anceaux

Supervisors: Asterios Katsifodimos, Andra Ionescu

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Duyemo Anceaux
Final project course: CSE3000 Research Project
Thesis committee: Asterios Katsifodimos, Andra Ionescu, Elvin Isufi

Abstract

Since every day more and more data is collected, it becomes more and more expensive to process. To reduce these costs, you can use dimensionality reduction to reduce the number of features per instance in a given dataset.

In this paper, we will compare four possible methods of dimensionality reduction. The feature extraction methods PCA, LDA, and GDA, and the feature selection method Lasso. We will mainly be comparing how the amount of features left over by these methods affects the accuracy of certain classification algorithms, and how long the methods take to achieve their task.

Our research highlights LDA as a highly effective method for significantly reducing the dimensionality of data used in logistic regression and Support Vector Machines (SVMs) with remarkable success. Additionally, we identified Lasso as the preferred choice for situations involving a limited training dataset or when utilizing the random forest algorithm for classification. Notably, Principal Component Analysis (PCA) was observed to occupy a middle ground between LDA's strengths in aggressive data reduction and Lasso's accuracy while retaining. GDA (with a linear kernel function) turned out to be significantly slower than the other methods, while its results where most of the time on par with LDA.

.

1 Introduction

The exponential growth of available data has necessitated effective strategies for processing and analysis. However, the increasing volume of data, coupled with the curse of dimensionality, presents significant challenges in terms of processing costs, both in terms of memory and time. To alleviate these concerns, dimensionality reduction techniques have emerged as valuable tools to reduce the number of features per object in a dataset. This research aims to compare the effects of several prominent dimensionality reduction methods, including Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Generalized Discriminant Analysis (GDA), and Least absolute shrinkage and selection operator (Lasso). The objective is to facilitate informed decision-making in selecting the appropriate method for dimensionality reduction in specific scenarios.

Dimensionality reduction encompasses a broad range of methods to decrease the number of features in objects of an original dataset. Among these methods are feature selection techniques, which involve choosing a subset of the original features. Lasso serves as an illustrative example of a feature selection method [1]. Conversely, alternative techniques employ feature extraction to generate new features that encapsulate the most pertinent information from the original dataset. The methods of feature extraction discused in this paper are PCA, LDA, and GDA [2]. These techniques often retain more information per feature compared to feature selection methods[3]. However, a drawback of feature extraction methods lies in the potential obscurity regarding the extracted features within the new reduced-dimensional datasets [4].

To conduct an intensive comparison of the selected dimensionality reduction methods, multiple aspects will be examined. Firstly, the impact of the number of retained features on the accuracy of three algorithms, Logistic Regression, Random Forest, and Support Vector Machines (SVMs), will be evaluated. The number of features to be retained can be specified for PCA, LDA, and GDA, while Lasso allows the adjustment of a hyperparameter indirectly influencing the number of features preserved. Additionally, the computational time required by each method to transform the original data into a new dataset with reduced dimensionality will be measured.

Recognizing that no single method universally outperforms others across all datasets [5] [6], this research will encompass the evaluation of these methods on multiple datasets. These datasets will vary in characteristics such as the number of training instances, the number of features, and the number of classes. The objective is to provide a comprehensive overview of how each algorithm behaves under diverse circumstances. This comparative analysis aims to equip researchers in the future with a clearer understanding of the advantages and limitations associated with each method. Thus, they can make more informed decisions in selecting the most suitable approach for specific instances. That is why our research question is: What are the effects of PCA, LDA, GDA, and Lasso for dimensionality reduction on classification algorithms?

2 Related work

Previous studies have conducted comparisons between PCA, LDA, and GDA. In a study focused on handprint recognition, GDA was found to significantly outperform PCA and LDA, albeit using a limited number of training and testing examples per class (5 each) and a relatively high number of classes (86) [2]. The specific classification algorithm employed in this study, however, remains unspecified. These findings suggest that GDA may yield superior results compared to other algorithms.

Another investigation explored the impact of various dimensionality reduction methods, including PCA and Lasso, on the classification accuracy of multiple datasets [5]. This study specifically examined the effects on SVM and Random Forest classification algorithms. The results indicated that Lasso generally outperformed PCA in terms of accuracy when using Random Forest, while for SVM, PCA showed the potential to outperform Lasso for certain datasets. These findings emphasize the importance of comparing different dimensionality reduction methods across various classification algorithms.

Furthermore, a separate study highlighted the influence of training data size on the relative performance of PCA and LDA [6]. While LDA generally outperformed PCA in terms of accuracy, it was observed that PCA could outperform LDA when the training set size was sufficiently small. These results show the significance of considering the training dataset size as a factor when determining the optimal algorithm for a given dataset.

Notably, limited studies directly compare the classification accuracy after employing LDA and Lasso or GDA and Lasso. This stems from the fact that Lasso primarily performs feature selection, while LDA and GDA focus on feature extraction. Moreover, existing studies that incorporate both LDA and Lasso often employ them for diffrent purposes, such as using Lasso for dimensionality reduction and LDA for classification [7]. However, studies comparing feature extraction and feature selection methods have established that neither approach universally outperforms the other. Feature selection excels at eliminating redundant features, while feature extraction is more effective in discriminating between classes, albeit potentially incurring higher computational costs. It is also worth noting that data resulting from feature selection remains interpretable, whereas data derived from feature extraction may lack interpretability [3; 4].

Given the lack of concrete comparisons between Lasso, GDA, and LDA for dimensionality reduction, conducting a comprehensive study would provide valuable insights into their relative performances. Such research would offer clarity regarding the optimal method to employ for datasets in various scenarios. Furthermore, while the aforementioned algorithms have been compared to PCA in different contexts, including PCA in the comparison framework of this study would establish a benchmark, as PCA is widely utilized for dimensionality reduction [3].

3 Preliminaries

In this section, we provide a brief description of the algorithms used in this comparative study. We begin by discussing the dimensional reduction methods, followed by a description of the three classification algorithms employed.

3.1 Dimensional reduction methods

In this section, we will shortly explain the workings of the dimensional reduction methods used in this comparison study.

PCA

PCA stands for principal component analysis [2]. The goal of PCA is to find the directions of maximum variance in the original dataset, orthogonal to each other. Then it chooses n of these directions the explain the most variance and projects the old data onto these new directions. For this algorithm, you can either directly specify how many features to have in the new data, or choose a percentage of the variance of the original dataset that needs to be explained in the new reduced dataset.

LDA

LDA stands for Linear Discriminant Analysis [2]. The goal of LDA is to find a linear projection that maximizes the ratio of between-class scatter to within-class scatter. One important consideration when using this method is that number of features in the resulting projection is constrained by the number of distinct classes in the original dataset minus one.

GDA

GDA stands for Generalized Discriminant Analysis [8]. Similarly to LDA, the goal of GDA is to find a projection that maximizes the ratio of between-class scatter to within-class scatter, and the number of features in the resulting projection is constrained by the number of distinct classes in the original dataset minus one. But GDA extends this method by having a unique covariance matrix for each class. Furthermore, while LDA assumes linear separability, GDA can extend this concept further by incorporating kernel functions.

Lasso

The last method for dimensional reduction that will be discussed is the Lasso algorithm. Lasso stands for least absolute shrinkage and selection operator [1]. Lasso is a linear model. Linear models try to model a target feature as a linear combination of all available features. To find the optimal weight for all features linear regression is used. Lasso then extends the principle of linear regression with the l_1 penalty, the penalizes the total weight of all features combined. A parameter α is used to denote how heavy the penalty is. So generally, but not always, the higher the α parameter is set, the fewer features are left in the dataset.

3.2 Classification algorithms

In this part of the paper, we will give a brief overview of the three classification algorithms that will be used in the research. These algorithms will be used to test the reduction of the accuracy due to the dimensional reduction algorithms.

Logistic regression

Logistic regression for multi-class classification works by using a set of binary logistic regression models, one for each class, to estimate the probabilities of each class independently. Each binary logistic regression model predicts the probability of belonging to a specific class versus the probability of belonging to the rest of the classes. The final class assignment is then determined based on the highest probability among all the classes.

Random Forest

Random forest for multi-class classification works by creating multiple decision trees, where each tree is trained on a bootstrap sample of the data and uses a random subset of features. During prediction, each tree independently assigns a class label, and the final class assignment is determined by majority voting among all the trees. The random forest algorithm is capable of handling multi-class classification by extending the majority voting mechanism to support multiple classes

SVM

Support Vector Machines (SVM) for multi-class classification work by transforming the original input space into a higher-dimensional feature space using a kernel function. The SVM algorithm finds the optimal hyperplane that maximizes the margin between different classes in this transformed space. During prediction, the class label is assigned based on the position of the test sample relative to the decision boundaries.

4 Methodology

This part will be split into 2 subsections. The section will give a quick overview of the datasets used in this research, and explain why they were selected. The second part explains how the data will then be preprocessed.

4.1 Datasets

In this subsection, we will give a short overview of the datasets that will be used in this research. Table 1 gives a short overview of all datasets used and their most important attributes. Below a short explanation of the different datasets will be given.

BCWD

The Breast Cancer Wisconsin (Diagnostic) Data Set [9], or BCWD for short, is a dataset containing data about tumors. The dataset was assembled with the goal of binary classification, predicting if a given tumor is benign or malignant given its features. This dataset was chosen because we wanted to see the results of LDA and GDA for binary classification. This is due to the inherent limitations of the algorithms, which restrict them to extracting only one feature per instance in a binary dataset.

Character Font Images Data Set

The dataset used in this study initially comprises images from 153 character fonts [10]. Augmented versions of this dataset, namely Fonts-10, Fonts-10-small, and Fonts-20, are employed for the experiments. Fonts-10 and Fonts-10-small select 10 character fonts from the original dataset, while Fonts-20 selects 20 character fonts, allowing the investigation of the impact of varying class sizes. Fonts-10 contains 10,000 instances, and Fonts-20 contains 20,000 instances, ensuring a consistent average number of instances per class across these datasets. The distinction between Fonts-10 and Fonts-10-small is that Fonts-10-small comprises only 1,000 instances, serving as a means to examine the effects of a smaller training set.

MNIST

MNIST is a dataset made with the goal to classify digits based on their images[11]. To maintain consistency with the Fonts-10 dataset, an augmented version of the MNIST dataset containing 10,000 instances is utilized. This augmented version retains the same number of classes as the Fonts-10 dataset but exhibits a feature set with higher dimensionality. The rationale behind selecting this dataset is to examine how varying the number of features influences the classification results.

Crops

This dataset contains radar and optical features from pieces of land used to grow crops [12]. For this dataset to goal is to classify which type of crop is grown on a piece of land given these features. An augmented version of this dataset will be used for the experiments. In this augmentation, the number of instances in the dataset is shrunk to 10,000. This is done for the purpose of keeping the number of instances consistent with the Fonts-10 dataset. This dataset was chosen because it is not an image classification dataset, like the character font and MNIST datasets. We wanted to see if this would lead to any significant changes in the results.

4.2 Preprocessing

All datasets will be read and split into a training set and a test set. The test sets will get 10% of the data, while the training data gets 90%. An exception is for the Fonts-10-small dataset where 900 more test instances were added when testing, due to the test size becoming too small to get useful results. All features will be standardized by removing the mean from their value and then scaling them to unit variance. After that, all labels are turned into integers in the range [1..n] where n is the number of classes. Because of the way the GDA method is implemented, the matrixes containing the data need to be transposed before GDA can use the data.

5 Evaluation

To have a clear understanding of the evaluation of the code, it is important to understand the metrics that will be used to evaluate the code. There are 3 main metrics we will be looking at when evaluating the code. The first metric is accuracy. Since all algorithms used to measure the effects of the dimensional reduction methods are used for classification. Because of that, the accuracy will represent the percentage of test cases correctly classified by the algorithms. The second metric is the dimensionality of the data after dimensional reduction is used. We want to compare how the dimensionality (amount of features) of the new dataset influences the accuracy of the classification algorithms. Lastly, we will look at how long it takes each of the algorithms to apply the dimensional reduction algorithm on a given dataset.

5.1 Experimental setup

For the experiments, we want to look at how the number of features in the new dataset, after using dimensionality reduction, affects the accuracy of the classification algorithms. For the sake of transparency, we will explain how the point at which the number of features and accuracy are measured are chosen.
For PCA, the number of features to keep must be in the range [1..n-1], where n is the number of features in the original dataset. Because this range can become big fast for datasets with many features, you can instead specify the percentage of the variance of the original dataset you want at least to see explained by the new dataset. In that case, the algorithm will pick the minimum number of required features to fulfill this criterion. In the experiments, this value will be used to change the number of features retained in each iteration of the algorithm. This value will start at 0.05 and will increase in steps of 0.05 up to and including 0.95.

For LDA and GDA the number of features in the new dataset is in the range is [1..c-1], where c is the number of classes of the dataset. Because there is a limited number of dimensions LDA and GDA can retain for the datasets we plan to use, we will just look at all possible numbers of dimensions to retain for both of them.

For the Lasso algorithm, there is a hyperparameter α that can be adjusted to influence the number of features selected by the algorithm. α can theoretically be any float in the range

Datasets			
characteristics \ datasets	#classes	#features	#instances
BCDW	2	30	569
Fonts-10	10	412	10,000
Fonts-10-small	10	412	1,000
Fonts-20	20	412	20,000
MNIST	10	784	10,000
Crops	7	175	10,000

Table 1: Used datasets and there characteristics

[0..inf). However, the higher α becomes, the more features it will eliminate. When α becomes too high, no features will be left. Obviously, we can not run any classification algorithms on data with zero features. Thus α has a limited range in practice. In the experiments, we are going to run, α will be varied from 0.01 up to and including 0.4, in steps of 0.01, or until α becomes too big and no features are selected.

The goal of varying these parameters is to give a clear picture of the influence of the dimensionality left by an algorithm on the accuracy of the classification algorithms.

After having a picture of this, it will become possible to compare which method of dimensionality reduction would be preferred in which situation. To make this comparison we will also look at the time it takes the algorithms to perform dimensionality reduction on the datasets. To make the process of testing this as unbiased as possible, all methods will be run on the same system, with as many background programs disabled as possible. This is to ensure as little as possible external factors may influence the processing time of the dimensionality reduction.

all code and data used can be found in the following git repository: https://github.com/delftdata/bsc_research_project_q4_2023.git

5.2 Results

This section will be split into 3 subsections. In the first subsection, we will look at the effect that the number of dimensions left by an algorithm has on the accuracy of the three classification algorithms for the datasets. In the second section, we will take a look at how long it takes each of the algorithms to perform dimensional reduction on a given dataset. In the third section, we will compare the effects of the dimensionality reduction methods on the accuracy of the classification algorithms.

accuracy vs features

In this section, we will show the effect of the number of features left in a dataset on the accuracy of the three classification algorithms. This will be done with the use of graphs. Graphs will show on the horizontal axes the number of features left and on the vertical axes the accuracy of the algorithms. The algorithm's names will be shortened in the legends of the graphs. Logistic regression will become 'LR', Random forest becomes 'RF', and support vector machine becomes 'SVM'. Keep in mind that the horizontal scales of the graph can greatly differ between dimensionality reduction methods because the maximum number of dimensions kept greatly differs between algorithms as mentioned in section 5.1. Since showing all graphs for all datasets is not feasible only certain graphs are selected.

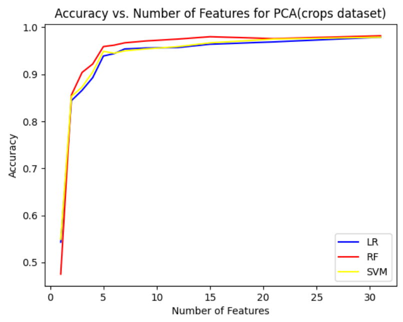

Figure 1: Accuracy vs #features using PCA for Crops dataset

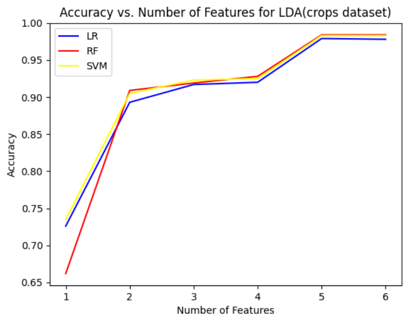

Figure 2: Accuracy vs #features using LDA for Crops dataset

The first results we will show are those of the crops dataset. As you can see in Figure 1 and Figure 2, for both the PCA and LDA algorithms, the classification algorithms behave mostly the same for a given algorithm. The graph of GDA is almost identical the same as the one of LDA and therefore left out. The graph of Lasso is almost the same as PCA and therefore left out. All 4 of these algorithms seem to all at first increase in accuracy a lot when adding more features, but then start to increase less and less the more features are added, converging to a certain accuracy. The best-performing instances of each combination of these 3 dimensionality reduction algorithms and the classification algorithms fall in a range of

[0.979, 0.988]. This is a small range. Important though, is that if you use PCA, you need around 12 features to reach this range of accuracy. LDA and GDA only require 5 features to reach this range. Lasso needs between 13 and 23 to reach this range depending on the classification algorithm.

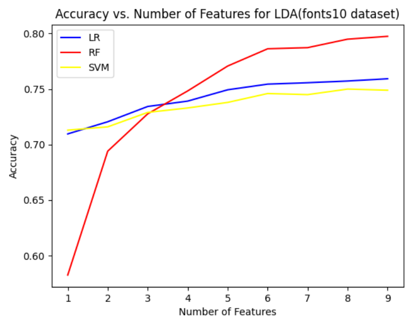

Figure 3: Accuracy vs #features using LDA for MNIST dataset

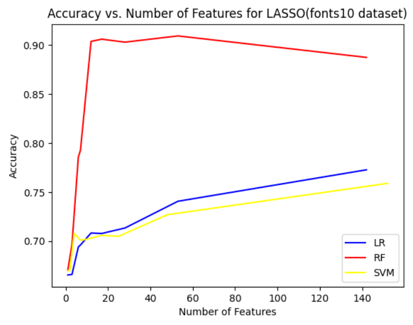

Figure 4: Accuracy vs #features using Lasso for Fonts-10 dataset

Next, we will discuss the results of the Fonts-10 dataset. These results are shown in Figure 3 and Figure 4. The graph of GDA is not shown since it is similar to that of LDA. Similarly, the graph for PCA is not shown since it is similar to the one of Lasso. On this particular dataset, the RF classification algorithm seems to be significantly more efficient than the other two. You can see also the Lasso algorithms combined with the RF algorithm give a significantly higher accuracy than other algorithms, even with a small number of features. RF behaves differently from the LR and SVM algorithms, which generally seem to increase in accuracy when the amount of features goes up. These 2 algorithms seem to behave consistently across all algorithms, although their accuracy is significantly worse when using PCA compared to the other 3 algorithms. Furthermore, you can see that LDA needs fewer features to reach similar accuracy for LR and SVM. But eventually, Lasso is able to outperform LDA, but with significantly more features, and not by much. For example. When Lasso gives 142 features, the accuracy of the LR algorithm becomes 0.773, while LDA with just 9 features has an accuracy of 0.759. This is only a difference of 0.014 in accuracy, for 133 features. Even when Lasso uses 53 features it only has an accuracy for the LR algorithm of 0.741, which is lower than the accuracy of LDA for 9 features.

The next dataset we tested was the Fonts-10-small dataset. When using this dataset, there does not seem to be as clear a correlation between the number of features and the accuracy of the algorithms. Figure 5 shows the effects of the number of features extracted by the GDA algorithm, on the accuracy of the classification algorithm. Since all graphs behave generally the same, we only put the graph for GDA here. The LDA graph looks a lot like the one of GDA, only there are the SVM scores relatively worse at any number of features. The graphs for PCA and Lasso are not here, since they look like their counterparts of the Fonts-10 dataset, only with more overfitting when you get to a higher amount of features. LDA and GDA seem to suffer much more from overfitting than PCA and Lasso. While the accuracy of the algorithms increases at first when Lasso and PCA allow for more features, this is not the case for LDA and GDA. The latter two get all their best scores in terms of accuracy when having either 1 or 2 features. All methods also are significantly less accurate on this dataset than on the standard Fonts-10 dataset.

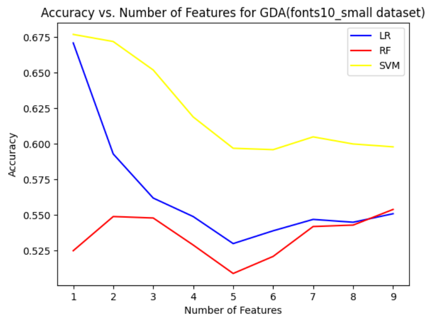

Figure 5: Accuracy vs #features using GDA for Fonts-10-small dataset

Now we are going to take the Fonts-20 dataset. The graphs of this dataset will not be shown, because these graphs look a lot like Figure 3 and Figure 4. These sets of graphs have two major differences, however. First of all, points in the graphs from the Fonts-20 dataset have less accuracy when compared to points with an equal amount of features on the other graph (around 0.15 to 0.25). Secondly, the LDA and GDA graphs of the Font-20 dataset have a wider range of features. Now you how 19, instead of just 9 features. But because each new feature added seems to add less and less accuracy, it does not really make a significant difference in the end. It is important to mention that making comparisons between points based on the number of features they have for the Lasso or PCA graphs, is not something that can be done precisely. This is because for Lasso and PCA we can not choose the number of features left after using it. To compare in the case of Lasso we mostly look at points that are generated using the same value for α. Except for the first two-point in the graph, the number of features over both datasets generated by choosing the same α never differ more than 1, in the case of these two datasets. When looking at differences between the PCa sets, You can compare points that are generated by having the same amount of original variance that needs to be presented in the new dataset. When we look at these pair from both graphs, they never differ more than 10%.

Now, let's look at the MNIST. We only took put one graph in the paper because all graphs for this dataset look the same (Figure 6. All algorithms seem to sharply increase in accuracy when more features are added at first. Then, after a sufficient amount of features, their accuracy starts to plateau. Lasso and PCA seem to slightly outperform GDA and LDA in terms of accuracy, but only when they start including more features than is possible for the latter two.

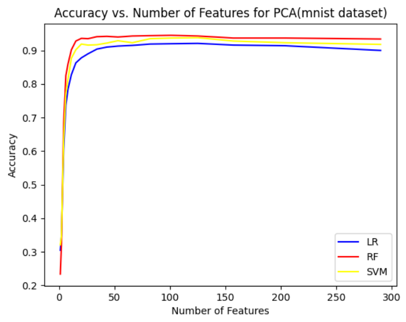

Figure 6: Accuracy vs #features using PCA for MNIST dataset

Lastly, we will look at the BCDW dataset. Since this dataset only has two classes there are no graphs to show for the LDA and GDA dimensionality reduction algorithms, since the only option these two algorithms have is the create one new feature. The PCA algorithm (Figure 7 shows that it only needs to generate a small number of features to become quite accurate. And you see that it generating more features can actually lower the accuracy of the classification algorithms. The graph for the algorithms after using Lasso peaks at a relatively low amount of features, and then falls off when it starts overfitting.

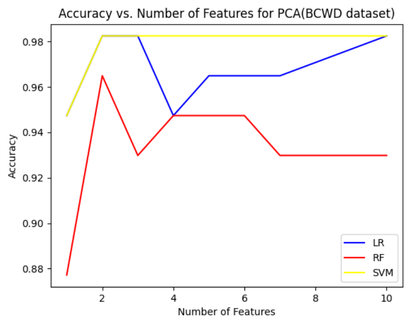

Figure 7: Accuracy vs #features using PCA for BCWD dataset

Speed of dimensionality reduction

For each of the datasets, we will test how long it takes each of the dimensionality reduction algorithms to change to transform the training dataset into a new dataset with a reduced number of features. This time will include both the time the fit the algorithms and to transform the data. Test data will not be taken into account in these experiments. The parameters for this experiment will be chosen as follows. For LDA and GDA the number of features in the new dataset will be the maximum amount of features they can create (the number of classes in the dataset minus one). For Lasso and PCA we will pick the parameter (either α or percentage of original variance to keep) that leads to a number of features in the new dataset that is as close as possible to the number of features left by the LDA and GDA algorithms. The results of the experiment are shown in Table 2.

As you can see in the table the GDA algorithm takes a very long time to complete compared to the other algorithms. When testing all stages of this algorithm it is clear to see that the step that takes the longest is the creation of a class-specific kernel for each instance in the dataset. This just takes a lot of time to do and naturally gets worse the more instances are in a dataset.

accuracy compared

Let us now take a closer look at how the accuracy of the algorithms actually compares to each other. We will take two datasets and make a table of those datasets showing 3 aspects for each combination of dimensionality and classification algorithm. The first aspect is the best accuracy they can be active while reducing the data to 10 features or less (which will be depicted by "acc¡10" in the tables). This is to show which algorithm would be best when severely reducing the number of features of a given dataset. Secondly, we will also show the best accuracy achieved by each of the algorithms (depicted by "best acc"). Lastly, we will show the number of features in the new dataset that scored the highest accuracy for each of the algorithms (depicted by "best #features)". This will then allow us to reason about potential trade-offs.

Firstly we will look at the results gathered from the Fonts-10 dataset. Most conclusions drawn from this dataset also apply to the Fonts-20 dataset, the MNIST dataset, and the Crops dataset, since they all behave relatively similarly. The only exception is that differences in accuracy between random forest and the other two classification algorithms are relatively smaller for the MNIST and Crops datasets. In Table 3 results you see clearly that LDA and GDA outperform Lasso and PCA when we only want to retain a very small amount of features (10 or less) when using either logistic regression or SVM as your classification algorithm. There are cases where PCA or Lasso can still obtain a higher accuracy than the GDA and LDA algorithms, but only when using significantly more features. This is however not the case when looking at the random forest classification algorithm. When using this algorithm PCA and Lasso seem to do equally well to GDA and LDA for relatively small amounts of features.

Now we will take a look at the results of the Fonts-10-small dataset (Table 4. There are two main takeaways from this graph. The first one is, that in this dataset with less training

Dimenionality reduction time				
Algorithm \ datasets	PCA	LDA	GDA	Lasso
BCDW	00:00.01	00:00.01	00:02.69	00:00.01
Crops	00:00.27	00:00.48	24:22:26	00:01.53
Fonts-10	00:00.89	00:01.38	49:13.65	00:01.15
Fonts-10-small	00:00.35	00:00.39	00:10.44	00:00.07
Fonts-20	00:01.52	00:02.37	01:38:13.03	00:40.48
MNIST	00:01.77	00:02.48	29:20.05	00:03.95

Table 2: Time it took the algorithm the reduce the dimensionality of datasets

data, PCA and Lasso now consistently manage to outperform LDA and GDA, even when using 10 or fewer features. You can also clearly see how all of the algorithms struggle with overfitting on this smaller dataset since the number of features at which they have the highest accuracy is significantly down from Table3.

6 Discussion and Limitations

6.1 Findings

In this section, we will name the most important findings of this experiment.

changes in the number of classes or features

When studying the difference between the Fonts-10, Fonts-20, Crops, and MNIST datasets, we noticed that the graphs of the data looked a lot alike. The only big difference was the overall accuracy of all classification algorithms after all dimensionality reduction algorithms were higher or lower depending on the dataset.

Overall patterns of the data however stayed the same. This leads us to conclude that when attempting classification using the algorithms of this paper, the number of classes and features do not significantly influence which combination is better than others, although these factors do influence the overall accuracy. This is in contrast to how the size of the training set actually influences which algorithms perform better, as shown in the next paragraph.

aggresive dimensionality reduction

The results obtained from our analysis show that when the objective is to reduce the number of features to below 10 while employing logistic regression or SVMs for classification, both LDA and GDA consistently outperformed Lasso and PCA algorithms.

Compared to that, in scenarios where a random forest is utilized as the classification algorithm, PCA and Lasso emerged as more suitable choices. These algorithms exhibited superior performance when the objective was to reduce the dimensionality of the dataset while employing Random Forest for classification.

LDA and GDA for binary classification

Initially, we held the expectation that GDA and LDA algorithms would exhibit subpar performance when applied to a binary dataset. This assumption stemmed from the limited ability of these algorithms to extract only a single feature per instance. However, contrary to our initial expectations, the application of GDA and LDA dimensionality reduction techniques on the BCWD dataset yielded relatively high accuracy scores for the classification algorithms. Notably, the minimum accuracy score achieved was 0.93. Remarkably, Logistic Regression and SVM algorithms demonstrated comparable accuracy to Lasso and PCA algorithms in this context, only scoring a lower accuracy when using random forest, where LDA and GDA perform subpar to Lasso and PCA most of the time.

LDA vs GDA

LDA and GDA (at least when using a linear kernel function) behave quite similarly for all the datasets that we tested. This in itself is not that surprising, since both methods start from the same basic idea. What is surprising is that the GDA, despite being a much more complex algorithm, did not significantly outperform LDA in providing more accurate results for any of the datasets. Even when using a linear kernel we still expected GDA to significantly outperform LDA because of the class-specific covariance matrixes. But on the contrary, GDA never managed to beat the accuracy of LDA for a given algorithm and a number of features by more than 0.02. LDA also never managed to beat GDA by a margin of more than 0.03. When you look at the time it takes to reduce the dimensionality of a dataset with each of the algorithms, LDA is dozens of times faster. This leads us to conclude that it would be better to use LDA than GDA, when a linear kernel function is used for GDA.

LDA vs PCA vs Lasso

In scenarios where the training dataset size is sufficiently large, the selection among LDA, PCA, and Lasso relies on two critical factors: the choice of the classification algorithm and the tradeoff between accuracy and the number of retained features in the transformed dataset. It is important to note that retaining a larger number of features comes at increased computational and storage costs.

LDA is generally the algorithm to choose when you want the new dataset to retain ten or fewer features. When wanting to focus more on accuracy, Lasso might be the best algorithm, if you use it to retain more features. If you want just a bit more accuracy but do not want to retain too many features, sometimes PCA is better, mostly between the 15 and 25 features.

Furthermore, when working with small training datasets, the Lasso algorithm demonstrated notable advantages over other approaches. It consistently delivered the best perfor-

	LR			RF			SVM		
	acc<10	best acc	best #features	acc<10	best acc	best #features	acc<10	best acc	best #features
PCA	0.667	0.736	110	0.798	0.809	19	0.670	0.732	110
LDA	0.759	0.759	9	0.798	0.798	9	0.750	0.750	8
GDA	0.745	0.745	7	0.762	0.762	7	0.752	0.752	8
Lasso	0.696	0.773	142	0.793	0.909	53	0.702	0.759	142

Table 3: Accuracy of the classification algorithms after using dimensionality reduction on the Fonts-10 dataset

	LR			RF			SVM		
	acc<10	best acc	best #features	acc<10	best acc	best #features	acc<10	best acc	best #features
PCA	0.676	0.678	15	0.684	0.685	18	0.676	0.689	39
LDA	0.671	0.671	1	0.550	0.550	2	0.637	0.637	1
GDA	0.671	0.671	1	0.549	0.549	2	0.677	0.677	1
Lasso	0.711	0.712	19	0.698	0.742	19	0.707	0.707	10

Table 4: Accuracy of the classification algorithms after using dimensionality reduction on the Fonts-10-small dataset

mance among all algorithms when using the Fonts-10-small dataset.

6.2 Limitations

A big limitation we ran into is that we could not find any Python libraries that implemented a version of the GDA algorithm. The only version we could find was written in Matlab. Since all other algorithms we needed were readily available in Pyhton we decided to write our code in Pyhton and use matlab.engine to call the Matlab code for the GDA algorithms in Pyhton. This, however, led to problems. For example, we could not run this code on the TU Delft cluster that was provided to us because we could not succeed in creating a Docker image with matlab.engine. Because we could not get this code running on the cluster we had to run it locally. Because of this, we had to limit the number of instances in datasets we used, because the time it would take to run this algorithm for bigger datasets was not feasible for this project.

Additionally, due to time constraints in the project and the size limitations of the paper, we were unable to explore multiple kernel functions for the GDA algorithm.

7 Conclusion and Future work

This section will first conclude the paper, and the briefly talk about potential future work

7.1 Conclusion

In this study, we investigated the impact of four dimensionality reduction algorithms on three distinct classification algorithms. Our findings demonstrate that there is no universally optimal dimensionality reduction algorithm, as certain algorithms exhibit superior performance in specific scenarios. Specifically, we established the advantages of Linear Discriminant Analysis (LDA) over Generalized Discriminant Analysis (GDA) when employing a linear kernel function.

Our research highlights LDA as a highly effective method for significantly reducing data dimensions in logistic regression and Support Vector Machines (SVMs) with remarkable success. Additionally, we identified Lasso as the preferred choice for situations involving a limited training dataset or when utilizing the random forest algorithm for classification. Notably, Principal Component Analysis (PCA) was observed to occupy a middle ground between LDA's strengths in aggressive data reduction and Lasso's accuracy while retaining greater numbers of features.

Moreover, our study revealed the relatively slow performance of the GDA algorithm in comparison to the other three algorithms analyzed.

Overall, these findings emphasize the importance of selecting the most suitable dimensionality reduction algorithm based on the specific requirements and characteristics of the classification task at hand

7.2 Future work

Two of the methods we research in this report make use of kernel functions. The dimensionality reduction method GDA, and the classification algorithm SVM. For both of these algorithms, there are multiple kernel functions that can be used in these functions. In this research, we only made use of linear kernel functions. In the feature, people could do research on the effects different kernel functions would have on these functions. Especially future researchers could compare GDA with different kernel functions to itself and the other classification algorithms.

Also, further research could be done on the effects of these dimensionality reduction methods on other classification algorithms. Algorithms like Convolutional Neural Networks are currently quite popular for image classification, and it would be interesting to see how the dimensionality reduction methods from this paper would influence an algorithm like that.

8 Responsible Research

All datasets used in this report were donated to the UCI Machine Learning Repository and references were made to either the repository itself or a specific paper if the provider of the paper so desired. The data itself did not give any ethical problems. All datasets we used were either non-sensitive, anonymous, or both.

The algorithm for GDA was made by Mohammad Haghighat and Saman Zonouz and Mohamed Abdel-Mottaleb [8]

The research done in this paper should be reproducible. A link to the GitHub repository containing the code of this project is provided in the paper. On the repository are also links to all the datasets. Since they are freely available this should not lead to any issues. Most datasets were too big to upload on GitHub, so are not there.

9 Acknowledgements

We wanted to send out our thanks to our supervisor Andra Ionescu for always being there for the help we needed it. We also wanted to thank her for always reacting quickly when we had a question.

We also want to thank Asterios Katsifodimos for supervising our project and giving us an understanding of how it is to conduct research.

References

[1] V. Fonti and E. Belitser, “Feature selection using lasso,” *VU Amsterdam research paper in business analytics*, vol. 30, pp. 1–25, 2017.

[2] P. Yu, P. Yu, and D. Xu, “Comparison of pca, lda and gda for palmprint verification,” in *2010 International Conference on Information, Networking and Automation (ICINA)*, vol. 1, 2010, pp. V1–148–V1–152.

[3] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, “A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56 – 70, May 2020. [Online]. Available: https://jastt.org/index.php/jasttpath/article/view/24

[4] Z. M. Hira and D. F. Gillies, “A review of feature selection and feature extraction methods applied on microarray data,” *Advances in bioinformatics*, vol. 2015, 2015.

[5] A. Babjac, T. Royalty, A. D. Steen, and S. J. Emrich, “A comparison of dimensionality reduction methods for large biological data.” New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi-org.tudelft.idm.oclc.org/10.1145/3535508.3545536

[6] A. Martinez and A. Kak, “Pca versus lda,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[7] D. C. Rich, K. M. Livingston, and S. L. Morgan, “Evaluating performance of lasso relative to pca and lda to classify dyes on fibers,” *Forensic Chemistry*, vol. 18, p. 100213, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2468170920300011

[8] M. Haghighat, S. Zonouz, and M. Abdel-Mottaleb, “Cloudid: Trustworthy cloud-based and cross-enterprise biometric identification,” *Expert Systems with Applications*, vol. 42, no. 21, pp. 7905–7916, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417415004273

[9] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, “Breast cancer diagnosis and prognosis via linear programming,” *Operations Research*, vol. 43, no. 4, pp. 570–577, 1995. [Online]. Available: https://doi.org/10.1287/opre.43.4.570

[10] R. Lyman, “Character Font Images,” UCI Machine Learning Repository, 2016, DOI: https://doi.org/10.24432/C5X61Q.

[11] L. Deng, “The mnist database of handwritten digit images for machine learning research,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[12] “Crop mapping using fused optical-radar data set,” UCI Machine Learning Repository, 2020, DOI: https://doi.org/10.24432/C5G89D.