Case Study: Econometrics and Data Science

Cancer subtype classification using shrinking methods on gene expression data

February 7, 2023

Students:
Jacco Broere

Caspar Hentenaar

Bas Willemsen

Abstract

This study investigates the impact of various dimensionality reduction techniques on classifying cancer subtypes based on gene expression data. In particular, the popular technique Principal Component Analysis (PCA) is compared to a Sparse version of PCA (SPCA) and a gene-expression specific sparse version of PCA (G-SPCA), both proposed by Zou et al. (2006). Data was obtained from four different sources, all concerning gene expression data for different subtypes of cancer. Results show that the percentage explained variance is positively related to the percentage of non-zero loadings for both SPCA and G-SPCA, with G-SPCA explaining a bit more variance given a certain percentage of non-zero loadings. Performance of subsequent classification algorithms is best when using regular PCA, winning or tying the classification accuracy for three out of four datasets. Runtime analysis shows that G-SPCA and PCA are significantly faster than SPCA, which is found to be exceptionally slow due to its increased computational complexity, making it not a viable option when dealing with extremely high-dimensional data such as gene expression data.

Contents

1 Introduction

1.1 Cancer

Cancer is a leading cause of death worldwide, as stated by the World Health Organization (WHO). In 2020, almost 10 million people died from cancer, representing about one out of six deaths globally. The disease is caused by abnormal growth and division of cells in the body. These cells can spread to other parts of the body through the blood and lymph systems, forming tumours. Cancer can affect any part of the body, including the organs, bones, and skin.

There are many different types of cancer, each with its own set of symptoms, causes, and treatments. The most common types of cancer are lung, colorectal, stomach, liver, and breast cancer (WHO). The number of deaths varies by region and country, with higher rates in low- and middle-income countries. The amount of cancer cases worldwide is expected to continue to rise as the global population ages and cancer risk factors become more prevalent, such as tobacco use and obesity (Torre et al., 2016).

Advances in cancer research have led to improved treatments and therapies, such as chemotherapy, radiation therapy, and surgery. New developments in immunotherapy and targeted therapy also show promising results (Debela et al., 2021). However, for all cancer types, it holds that early detection and diagnosis are crucial for successful treatment and survival rates (Blandin Knight et al., 2017). Analysing gene expression data is one of the ways to more accurately classify the kind of cancer in a patient and increase the chances of successful, timely treatment.

1.2 Gene expression data

Gene expression data can be used to classify different types of cancer and to understand the underlying molecular mechanisms of the disease. The process of gene expression involves the transcription of genetic information from DNA to RNA, and the subsequent translation of RNA into proteins. By measuring the levels of gene expression in cancerous tissue, researchers can identify patterns of altered gene activity that are associated with specific types of tumours.

Various methods for measuring gene expression exist, each with its own advantages and disadvantages. The choice of method depends on the specific research question and the type of sample being analysed. Examples of methods include microarrays, RNA-sequencing, real-time PCR and proteomics, which are left to the interested reader to explore further.

One way gene expression data is used in cancer classification is through the identification of biomarkers, which are genes or proteins that are expressed differently in cancerous tissue compared to normal tissue (Aguirre-Gamboa et al., 2013). These biomarkers can be used to distinguish between different types of cancer, and to classify the stage of the disease. For example, a specific pattern of gene expression in a breast cancer sample might indicate that the cancer is of a certain subtype and has higher likelihood of spreading (Lehmann & Pietenpol, 2014).

Another way gene expression data is used in cancer classification is through the use of machine learning algorithms, which can analyse large amounts of gene expression data and identify patterns that are not visible to the human eye (Chen et al., 2016; Yuan et al., 2020). These algorithms can be trained to classify cancer samples based on their gene expression profiles, and can help to identify new subtypes of cancer.

One problem concerning the analysis of gene expression data is the problem of high-dimensionality. Due to the high number of genes and the (often) small number of samples in a cancer study, traditional algorithmic and statistical methods are often not valid or infeasible.

Several dimensionality reduction techniques are often applied to alleviate these issues, which will be the focus of this paper.

1.3 Dimensionality reduction

High dimensionality in gene expression data refers to the large number of genes that are measured in a single experiment. This can lead to a number of problems when fitting models, including but not limited to the curse of dimensionality, overfitting, and increased computational complexity (Van Der Maaten et al., 2009).

Dimensionality reduction or shrinking is a technique used to simplify high-dimensional data, such as gene expression data, by reducing the number of variables while retaining as much of the important information as possible. The goal of dimensionality reduction is to identify the most informative features in the data and to represent them in a lower-dimensional space. In the context of gene expression data, dimensionality reduction can be used to identify the genes that are most strongly associated with a particular disease or condition, such as in this case cancer. This can make the data easier to visualize and analyse, and can also improve the performance of machine learning algorithms.

There are several methods for dimensionality reduction. In this paper, we will focus on Principal Component Analysis (PCA) and its extension, Sparse Principal Component Analysis (SPCA). SPCA attempts to identify a sparse set of principal components that accounts for the majority of the variance within the data, while simultaneously maximizing the number of zero loadings in the identified components. This is done by solving an elastic net. Additionally, we explore a sparse PCA algorithm specifically tailored to gene expression data, namely G-SPCA, as proposed by Zou et al. (2006). G-SPCA has the same goal as SPCA, but instead of solving an elastic net takes an alternative *soft-thresholding* approach, making it more suitable for situations where the number of features, genes in this case, is many times larger than the number of samples. More detailed theoretical and mathematical explanation on these methods will be given in chapter 3.

1.4 Research question and hypothesis

The research questions for this paper can be formulated as follows:

How do different (Sparse) Principal Component Analysis variants affect the performance and explainability of classification models on gene expression data?

Zou et al. (2006) report on the relation between non-zero loadings and percentage of explained variance. Both SPCA and *simple thresholding* applied to PCA attain similar performance. Simple thresholding implies a threshold is set a priori and loadings smaller than this threshold are set to zero. Based on these findings, transforming the data using SPCA is not expected to improve classification results compared to PCA. However, we deem the interaction between SPCA and the classification algorithms worth investigating, since Zou et al. (2006) note that SPCA and PCA do in fact yield different non-zero loadings and different feature sets. Furthermore, SPCA can aid in the explainability of the importance of different features, thus finding even comparable performance to PCA might be indicative of a preferable case for SPCA.

Following, in Section 2 the data for the experiment is presented. After this, in Section 3 the methodology is given in which the relevant methods are described in more theoretical and methematical detail, and the experimental setup is outlined. In Section 4 the results are presented and discussed. Lastly in Section 5, the conclusion and discussion are presented.

2 Data

2.1 Overview

For this paper, four gene expression datasets were used. Below, an overview of the different datasets is given, together with a short summary of each paper associated with the dataset used.

Table 1: Descriptive statistics for each dataset

Author	Cancer type	Samples	Features	Classes	Class distribution (%)
Khan et al. (2001)	SRBCT	63	2,308	4	12.7, 36.5, 19.0, 31.7
Alon et al. (1999)	Colon	62	2,000	2	35.5, 64.5
Gravier et al. (2010)	Breast	168	2,905	2	66.1, 33.9
Sørlie et al. (2001)	Breast	85	456	5	16.5, 12.9, 15.3, 17.6, 37.6

2.2 Paper summaries

Khan et al. (2001) investigated the use of gene expression profiling and artificial neural networks (ANNs) for the classification and prediction of different types of "small, round, blue cell tumors" (SRBCTs). They found that ANNs can accurately classify different subtypes of cancer based on gene expression data and can also accurately predict the diagnosis of new patients. The results suggest that ANNs could be a valuable tool for the diagnosis and classification of cancer. The data used in this study is a subset of the data in the original study.

Alon et al. (1999) explored the gene expression patterns in normal and tumor colon tissues. The study used clustering analysis to identify broad patterns of gene expression in the different tissues. The results showed that the gene expression patterns in tumor colon tissues were distinct from those in normal colon tissues and that the gene expression profiles could be used to differentiate between the two. The results suggest that gene expression profiles could be a valuable tool for understanding the biology of colon tumors and for developing new diagnostic approaches for colon cancer.

Gravier et al. (2010) investigated the development of a DNA signature to predict the prognosis of small (so-called T1T2) node-negative breast cancer patients. The study used DNA microarray analysis to profile gene expression in breast tumors and identify a set of genes that were associated with patient outcome. The results showed that a subset of 7 genes was able to accurately predict the likelihood of distant metastasis in small node-negative breast cancer patients. The results suggest that these 7 genes can be used as a focus tool for individualized treatment decision-making and for improving patient outcomes.

Sørlie et al. (2001) analyze gene expression patterns in breast carcinomas to identify cancer subclasses and their clinical characteristics. The study used DNA microarray analysis to analyze tumors from a large group of patients and found two distinct subclasses of breast carcinomas based on gene expression. The results showed that these subclasses were associated with different clinical outcomes and could be used to predict the prognosis of individual patients. The study concluded that the gene expression patterns could provide important information for the diagnosis and treatment of breast cancer, and could lead to the development of new therapeutic strategies that are tailored to the specific subclasses of the disease.

3 Methodology

3.1 Dimensionality reduction

3.1.1 Principal Component Analysis

Principal Component Analysis (PCA) is a widely used statistical technique for the dimensionality reduction of large, complex datasets. It was first proposed by mathematician Karl Pearson (1901) and later further developed by Hotelling (1933). PCA is a versatile technique that can also be used for applications such as data visualization, feature extraction, noise reduction, and denoising of data.

The objective of PCA is to identify the underlying structure of the data by identifying the directions (components) in the data that explain the most variance. This is achieved by projecting the data onto a new set of axes, referred to as principal components. The first principal component corresponds to the direction in the data that explains the greatest amount of variance, while subsequent components correspond to directions that explain decreasing amounts of variance. See Figure 1 for a simple example.

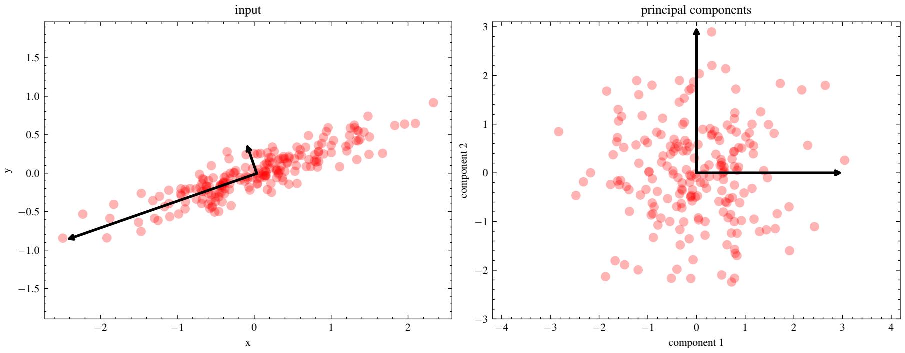


Figure 1: Visualization of breakdown of data into 2 principal components.

The implementation of PCA involves several steps. Firstly, the data must be standardized to ensure that all variables are on the same scale. This is a critical assumption of PCA. Let $\tilde{X} \in \mathbb{R}^{n\times p}$ be the matrix containing data on p features, consisting of n observations. Then construct the standardized data matrix $X \in \mathbb{R}^{n\times p}$, by transforming each feature $x_i \in \mathbb{R}^n$ with $i \in \{1, \dots, p\}$ using the following operation:

$$x_i = \frac{\tilde{x}_i - \hat{\mu}_{\tilde{x}_i}}{\hat{\sigma}_{\tilde{x}_i}}, \tag{1}$$

with $\hat{\mu}$ and $\hat{\sigma}$ denoting the sample mean and standard deviation respectively. Applying standardization ensures that the mean of each x_i is equal to zero, with a standard deviation of one.

Then, the covariance matrix $C \in \mathbb{R}^{p\times p}$ is given by $C = \frac{1}{n-1}X^TX$. C is a symmetric matrix and can be diagonalized as follows:

$$C = VLV^T \tag{2}$$

where V is the matrix containing the eigenvectors, and L is a matrix containing the eigenvalues λ_i in decreasing on the diagonal. The eigenvectors v_i are the principal axes, not to be confused with the principal components. The principal components themselves are the projections of the standardized data on the principal axes. The k principal components are given by the first k columns of XV.

However, PCA can also be performed by using singular value decomposition (SVD) on the standardized data matrix X:

$$\tilde{X} = USV^T, \tag{3}$$

where $U \in \mathbb{R}^{n \times n}$ is a unitary matrix and $S \in \mathbb{R}^{n \times p}$ is a diagonal matrix containing the singular values s_i. From this, we can conclude that

$$C = \frac{1}{n-1} VSU^T USV^T = V \frac{S^2}{n-1} V^T \tag{4}$$

which means that singular vectors V are the principal axes and the singular values are related to the eigenvalues of the covariance matrix C as $\lambda_i = s_i^2/(n-1)$. The principal components are given by $XV = USV^TV = US$. The sample variance of the ith principal component is equal to s_{ii}^2/n.

The output of PCA is the set of principal components, which are linear combinations of the original features, and are orthogonal to each other. One drawback of PCA is that each principal component is a linear combination of all p variables, which makes intuitive interpretation difficult. By pushing factor loadings to zero, the Sparse PCA method tries to explain most of the variance in the data while increasing interpretability.

3.1.2 Sparse Principal Component Analysis

Sparse Principal Component Analysis (SPCA) is a method that modifies the traditional Principal Component Analysis (PCA) by incorporating sparsity into the identification of principal components. SPCA attempts to identify a sparse set of principal components that can effectively account for the majority of the variance within the data, while simultaneously maximizing the number of zero loadings in the identified components. The goal of SPCA is to extract the most informative and relevant features from the data while reducing the dimensionality of the data and improving the interpretability of the results.

One of the main advantages of SPCA is that it can be used to identify a small set of important variables that are most strongly associated with the principal components, as opposed to being a linear combination of all variables as in PCA. This can be particularly useful in gene expression data, as SPCA can be used to identify a small set of genes that are most strongly associated with the concerning disease or condition. This sparse set of genes can subsequently be used for further analysis. Similarly, SPCA can be used to handle other forms of high-dimensional data, which are common in many real-world applications such as text data analysis (Zhang & Ghaoui, 2011).

There are multiple mathematical formulations of Sparse Principal Component Analysis (SPCA) each with its own advantages and disadvantages. We implement the SPCA algorithm as described in Zou et al., 2006:

1. Let A start at $V[:, 1:k]$, the first k loadings of the ordinary principal components

2. Given $A = [\alpha_1, \ldots, \alpha_k]$, solve the following elastic net problem for $j = 1, \ldots, k$

$$\beta_j = \arg\min_{\beta} (\alpha_j - \beta_j)^T (X^T X)(\alpha_j - \beta) + \lambda \|\beta_j\|^2 + \lambda_{1,j} \|\beta_j\|_1$$

3. given $B = [\beta_1, \ldots, \beta_2]$ compute $X^T X B = UDV^T$ using SVD, then update $A = UV^T$

4. repeat 2, 3 until convergence.

5. Normalize and return loadings $\hat{V}_j = \frac{\beta_j}{\|\beta_j\|}$

To solve the elastic net problem, we utilize the solver as implemented by *scikit-learn* (Pedregosa et al., 2011). Zou et al. (2006) note that the computational complexity of the elastic net regression is $O(p^3)$. Naturally, this SPCA method might be unfit when dealing with gene expression data, where $p \gg n$ is commonplace.

3.1.3 Sparse Principal Component Analysis for gene expression data

Sparse Principal Component Analysis for gene expression data (G-SPCA) is a modification of the SPCA procedure described in section 3.1.2, tailored specifically for data where the number of features is many times larger than the number of samples, as is often the case in gene expression data. Due to the high dimensionality of gene expression data, the Elastic Net problem in step 2. of section 3.1.2 can become infeasible due to the cubic time complexity. To circumvent this issue, Zou et al., 2006 propose an alternative *soft-thresholding* approach, by restricting the objective function in step 2. of the algorithm in section 3.1.2 as follows, for $j \in \{1, \dots, k\}$:

$$\hat{\beta}_j = \underset{\beta_j}{\arg\min} -2\alpha_j^T(X^TX)\beta_j + \left\|\beta_j\right\|^2 + \lambda_{1,j}\left\|\beta_j\right\|_1,$$

we are able to obtain an explicit solution given α_j, namely

$$\beta_j = \left(|\alpha_j^T X^T X| - \frac{\lambda_{1,j}}{2}\right)_+ \text{Sign}(\alpha_j^T X^T X). \tag{5}$$

The complexity of calculating $X^TX\beta$ is $O(p^2)$ for regular matrix multiplications. Note that we treat X^TX as given, as this matrix is constant for each iteration of the algorithm and therefore needs to be calculated only once during initialization. Clearly, a computational complexity of $O(p^2)$ (for G-SPCA) instead of $O(p^3)$ (for SPCA) for each iteration of the algorithm can be substantial when $p \gg n$, which is usually the case for gene expression data. However, the work of Zou et al., 2006 does not provide any guarantees for the number of iterations of the algorithm when utilizing this *soft-thresholding* SPCA. The efficiency of not conducting an elastic net regression is evaluated through runtime analysis to determine if the theoretical improvement in performance translates to a tangible improvement for practical applications.

Below, the soft threshold as utilized in equation (5) is visualized, alongside the *hard-threshold* counterpart.

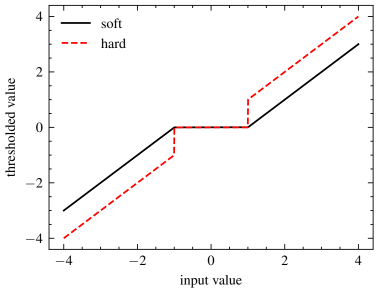


Figure 2: Soft and hard thresholding with threshold $\lambda = 1$

3.2 Classification algorithms

3.2.1 Logistic Principal Component Regression

Logistic Principal Component Regression (LPCR) is a variation of Principal Component Regression (PCR) that is used for categorical classification problems. As the name suggests, it combines the use of Principal Component Analysis (PCA) and Logistic Regression (LR) to build a predictive model. LPCR uses the principal components obtained from PCA as predictors in a logistic regression model, rather than using the original variables. This can help to reduce the problems of multicollinearity and overfitting that can occur in traditional logistic regression with high-dimensional datasets (Wold et al., 1984). LPCR can logically also be adapted to Logistic Sparse Principal Component Regression (LSPCR) or Logistic Gene-Sparse Principal Component Regression (LGSPCR) by using the components resulting from Sparse PCA or the soft-thresholding G-SPCA respectively as opposed to using the components from regular PCA.

The objective function for Logistic Regression for multi-class classification with K classes, s.t. $y_i \in \{1, \ldots, K\}$, can be formalized as (Anderson & Blair, 1982):

$$\widehat{W} = \arg\min_{W} -C \sum_{i=1}^{n} \sum_{j=0}^{K-1} I(y_i = j) \log \left(\frac{e^{X_i W_j}}{\sum_{l=0}^{K-1} e^{X_i W_l}} \right) + \frac{1-\rho}{2} \|W\|_F^2 + \rho \|W\|_1 \tag{6}$$

with W denoting the coefficient matrix, where each row W_k corresponds to class k. The hyperparameter $\rho \in [0, 1]$ controls the ratio between l_1- and l_2-regularization and $C \in \mathbb{R}_{>0}$ controls the general severity of regularization, where smaller values of C correspond to less regularization. The resulting $\widehat{W}$ matrix can then be used to predict probabilities of a new observation belonging to a particular class, the class with the maximum corresponding probability is then yielded as the prediction. The estimated probabilities for each class can be constructed as follows:

$$\hat{p}_k(X_i) = \frac{e^{X_i \widehat{W}_j}}{\sum_{l=0}^{K-1} e^{X_i \widehat{W}_l}} \tag{7}$$

3.2.2 Gradient Tree Boosting

Gradient Tree Boosting algorithms are a popular and powerful machine learning technique used for solving regression and classification problems. These algorithms build a model by combining multiple decision trees sequentially, where each tree aims to correct the mistakes made by the previous tree (Friedman, 2002). The trees are grown using the gradient of the loss function, explaining the nomenclature *Gradient Tree Boosting*. The final prediction is made by combining the predictions from all the trees using a weighted sum.

One of the most popular Gradient Tree Boosting algorithms is *LightGBM*. Developed by Ke et al. (2017), *LightGBM* offers several advantages over other tree boosting algorithms. Firstly, it uses a novel tree building algorithm called histogram-based tree building or Exclusive Feature Bundling (EFB). This technique divides continuous feature values into discrete bins and uses these bins to construct decision trees, resulting in faster training times as compared to traditional tree building algorithms which are based on sorting the feature values.

Secondly, *LightGBM* uses a gradient-based one-side sampling technique, which samples only the positive instances or negative instances with a higher probability. This results in faster convergence and improved accuracy. These improvements make *LightGBM* a highly efficient and scalable tree boosting algorithm that offers significant benefits over other algorithms.

3.2.3 Hyperparameter tuning

To optimize the performance of the classification models discussed in section 3.2.1 and 3.2.2, we employ *Optuna*, an open-source hyperparameter optimization framework (Akiba et al., 2019). *Optuna* addresses the limitations of traditional hyperparameter optimization methods by utilizing dynamically constructed parameter spaces. Furthermore, *Optuna* improves efficiency through efficient sampling and pruning mechanisms. Our implementation of *Optuna* uses the Tree-structured Parzen Estimator (TPE), which estimates two Gaussian Mixture Models (GMM) (Bergstra et al., 2011). Specifically, one Gaussian Mixture Model is fit to the parameters that result in the best evaluation score, which in this case is the classification accuracy over 3-fold cross validation within the training set (67% of the dataset). The other Gaussian Mixture Model is fit to the remaining parameters that are yet to be tried. Lastly, *Optuna* employs Asynchronous Successive Halving (ASHA) for efficient pruning, resulting in more cost-effective hyperparameter optimization (Li et al., 2018).

Table 2 presents the hyperparameter search spaces for both Logistic Regression (LR) and LightGBM (LGBM). *Optuna* uses this parameter space to sample hyperparameter settings iteratively for 50 iterations, using the procedure described above.

Table 2: Hyperparameter space for *LR* and *LGBM*

Hyperparameter	*LR*	*LGBM*
l1-ratio	$[0, 1]$	-
C	$[0.01, 1]$	-
Number of trees	-	100
Learning rate	-	0.1
Number of leaves	-	$\mathbb{N} \cap [15, 100]$
Maximum depth	-	$\mathbb{N} \cap [3, 20]$
Minimum samples in child node	-	$\mathbb{N} \cap [2, 5]$

Note: *Learning rate* and *Number of trees* are not optimized but set as a fixed value manually

3.3 Experiment outline

The experiments in this paper are outlined to accommodate a comparison of sparse formulations of Principal Component Analysis, with an application on gene expression datasets concerning cancer subtypes. Firstly, we perform experiments to compare the methods discussed in section 3.1.2 and 3.1.3 directly. Secondly, we make a comparison of these methods by using their output as a data preprocessing step for a classification algorithm which tries to predict the relevant cancer subtype in each of the datasets. To replicate relevant practical applications in which these methods can be used. Below, each of the necessary steps is outlined in more detail.

3.3.1 Preprocessing

First, all datasets are transformed into a suitable tabular format. This process consists of renaming the column names, transposing the dataset if necessary, dropping incomplete or duplicate observations and features, and adding the targets variables, specifically cancer subtypes in this application. Standardizing the data is necessary for all three formulations of PCA, as explained in section 3.1.1.

3.3.2 PCA, SPCA and G-SPCA

We apply the regular Principal Components Analysis (PCA), Sparse Principal Components Analysis (SPCA) and the soft-thresholding Gene-Sparse Principal Component Analysis (G-SPCA) as described in 3.1.3. For all three methods, the first 15 components are used. Moreover, for both SPCA variants, regularization parameters are set such that the methods yield a roughly equal amount of non-zero loadings to allow for fair comparison. PCA functions were implemented from the *scikit-learn* library in Python (Pedregosa et al., 2011). We implemented SPCA and G-SPCA ourselves, as the implementations available from existing libraries were either missing or insufficient for our experiments. The algorithms are implemented as described in section 3.1.

3.3.3 Classification

The three sets of principal components resulting from PCA, SPCA, and G-SPCA are utilized as features in the two classification algorithms, Logistic Regression (LR) and LightGBM (LGBM). The predictive performance of the classification algorithms, when trained on the three sets of principal components, are analysed and compared. Given the imbalanced nature of the two binary datasets, the macro F1-score is employed as the performance metric, as it assigns equal weight to both classes. In addition, the accuracy is reported because of ease of interpretability.

3.4 Evaluation

The evaluation of the proposed data transformations will be threefold, namely by the percentage of explained variance (PEV), the total runtime of each of the dimensionality reduction methods, and performance of the classification methods using the different principal component sets.

3.4.1 Classification performance

While Sparse PCA methods increase the interpretability of dimensionality reduction, we are also interested in how the performance of classification algorithms are affected by the different sets of principal components. As mentioned in Section 3.2.3, hyperparameter tuning for both classification models was done using *Optuna* to find the optimal hyperparameters, as in real world applications one would do. For both classification methods described in Section 3.2, the accuracy score and the macro-averaged F1-score is presented per dataset, given the different dimensionality reduction component results.

3.4.2 Percentage Explained Variance

A popular statistic to judge the performance of dimensionality reduction techniques is the percentage explained variance (PEV). Zou et al. (2006) provide the PEV of the first (sparse) principal component versus the amount of non-zero loadings. They use a method based on the QR decomposition to compute the variance captured by SPCA. Camacho et al. (2020) show that this method is generally incorrect, as the total variance calculated this way usually does not coincide with the actual total variance. Instead, we calculate PEV as in Guerra-Urzola et al. (2021):

$$PEV = 1 - \frac{\|\widehat{X} - X\|_F^2}{\|X\|_F^2}$$

Where $\widehat{X}$ denotes the recovered dataset obtained by multiplying the transformed dataset with the transposed loadings and $\|\cdot\|_F$ denotes the Frobenius norm. This way, the PEV measure captures the explained variance across all components in comparison to the entire dataset, instead of the leading PC in comparison to all components. For our methods, we are interested in the PEV compared to the number of non-zero loadings. By varying the regularization parameters in SPCA and G-SPCA, the amount of non-zero loadings can be chosen, which should have a negative relationship to the PEV. In the results for each dataset, for each dimensionality reduction method, the percentage of explained variance is plotted versus the percentage of non-zero loadings.

3.4.3 Runtime analysis

Lastly, we provide the runtime of each dimensionality reduction method. In order to make fair comparisons, we ensure that for each dataset the regularization parameter is set such that the percentage of zero loadings is equal up to a 2% difference between SPCA and G-SPCA. To achieve this α is set to 0.01 for SPCA and a bisection search over λ_1 is performed for G-SPCA to find the corresponding value. The runtimes are evaluated using $k = 5$ components and for each dataset, transformation combination the dimensionality reduction is performed 3 times. In the Results section, runtime tables can be found for the different combinations.

4 Results

4.1 Classification performance

In Table 3 and 4 the accuracy scores and macro-averaged F1-scores can be seen for all different combinations of dataset, dimensionality reduction method and classification model. PCA wins or ties for best performance in terms of accuracy and F1-score on three out of the four datasets. G-SPCA wins or ties two times in terms of accuracy and once in terms of F1-score. SPCA ties once for both accuracy and F1-score on the dataset from Sørlie et al. (2001).

As for comparing the different classification models per dimensionality reduction method, it can be seen that Logistic Regression (LR) outperforms LightGBM (LGBM) on three out of four datasets for PCA and SPCA. Utilizing G-SPCA components, LGBM outperforms LR on all 4 datasets.

Table 3: Accuracy score for *LR* and *LGBM*

	PCA		SPCA		G-SPCA	
	LGBM	LR	LGBM	LR	LGBM	LR
Sørlie et al. (2001)	0.79	**0.86**	0.72	**0.86**	0.83	0.79
Khan et al. (2001)	0.86	**1.00**	0.90	0.95	0.81	0.71
Alon et al. (1999)	0.71	0.86	0.86	0.86	**0.90**	0.67
Gravier et al. (2010)	**0.73**	0.71	0.64	0.71	**0.73**	0.71

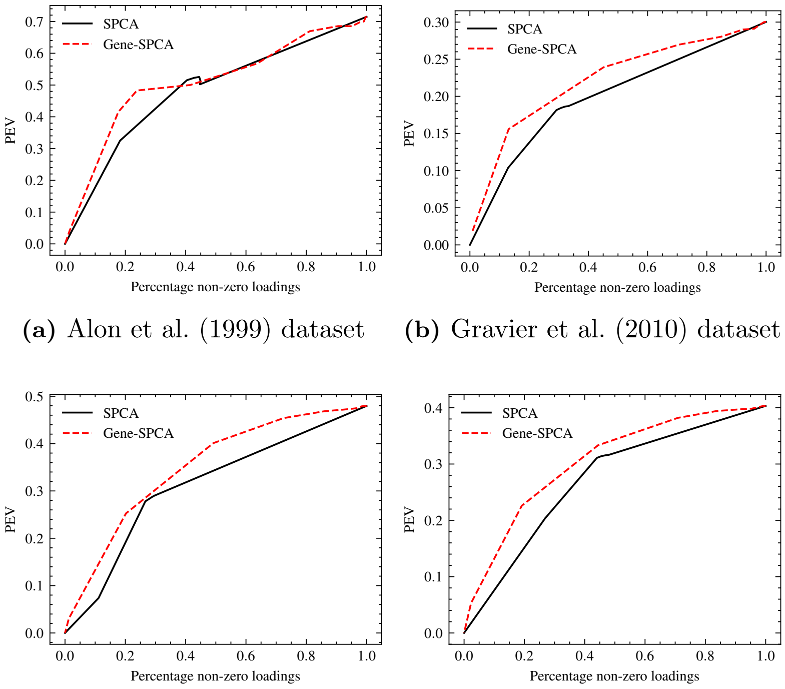
Table 4: Macro-averaged F1-score for *LR* and *LGBM*

	PCA		SPCA		G-SPCA	
	LGBM	LR	LGBM	LR	LGBM	LR
Sørlie et al. (2001)	0.79	**0.86**	0.72	**0.86**	0.81	0.74
Khan et al. (2001)	0.81	**1.00**	0.84	0.97	0.88	0.78
Alon et al. (1999)	0.68	0.85	0.84	0.85	**0.89**	0.40
Gravier et al. (2010)	**0.65**	0.58	0.60	0.58	0.61	0.58

4.2 Percentage Explained Variance

In Figure 3 the percentage explained variance for the four datasets can be seen for different percentages of non-zero loadings, for both SPCA and G-SPCA. SPCA is prone to numerical issues when using low regularization, therefore there are no observations between 50 and 100% of the non-zero loadings. SPCA usually achieves a slightly lower PEV in comparison to G-SPCA.

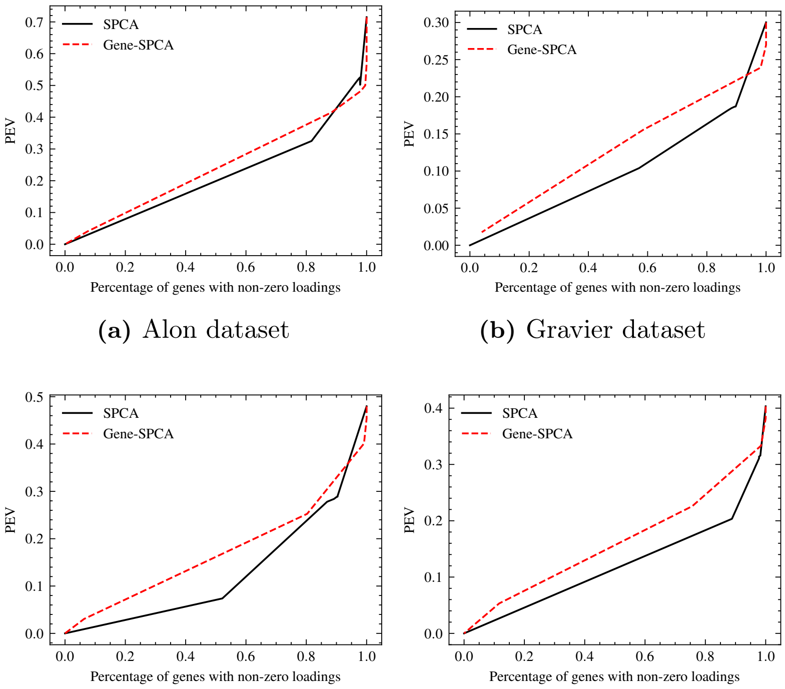
In figure 4 the PEV is plotted against the percentage of genes with non-zero loadings. A gene disappears from the set of principal components when the loadings for that gene are set to zero for all principal components. This is the case when an entire row in the loadings matrix is set to 0. Comparably to PEV vs percentage of non-zero loadings, SPCA generally attains a slightly lower percentage of explained variance for the same percentage of genes with non-zero loadings.



(a) Alon et al. (1999) dataset (b) Gravier et al. (2010) dataset

(c) Khan et al. (2001) dataset (d) Sørlie et al. (2001) dataset

Figure 3: Percentage of non-zero loadings vs the percentage of explained variance



(a) Alon dataset

(b) Gravier dataset

(c) Khan dataset

(d) Sorlie dataset

Figure 4: Percentage of genes with complete zero loadings vs the percentage of explained variance

4.3 Runtime analysis

Table 5: Runtime analysis of fitting transformation

		PCA	SPCA	G-SPCA
Khan et al. (2001)	$\bar{t}$	0.10	113.81	0.60
	σ_t	0.01	0.31	0.03
Alon et al. (1999)	$\bar{t}$	0.05	652.77	0.08
	σ_t	0.02	0.68	0.01
Gravier et al. (2010)	$\bar{t}$	0.08	119.07	1.98
	σ_t	0.06	0.21	0.11
Sørlie et al. (2001)	$\bar{t}$	0.04	54.40	0.05
	σ_t	0.04	0.44	0.02

Table 6: Student's t-test between runtime PCA and G-SPCA

	t-statistic	p-value
Khan et al. (2001)	-21.08	0.00
Alon et al. (1999)	-42.20	0.00
Gravier et al. (2010)	-38.50	0.00
Sørlie et al. (2001)	-6.18	0.00

In Table 5 the runtimes for the different dimensionality reduction methods are displayed, and Table 6 presents the results from the significance tests between PCA and G-SPCA are shown. For all datasets, G-SPCA and PCA compute faster than SPCA. Furthermore, PCA is significantly faster than G-SPCA at a 1% confidence level for all datasets.

5 Conclusion & Discussion

5.1 Runtimes

Results show that fitting PCA is significantly faster than G-SPCA on all datasets. This is expected as the first step of performing G-SPCA is performing PCA. Clearly, SPCA runtimes are significantly longer than for both PCA and G-SPCA. This is due to its increased computational complexity stemming from the elastic net algorithm, making it not a viable option when dealing with extremely high-dimensional data such as gene-expression data.

5.2 Explained variance

The results demonstrate that the SPCA generally produced a lower PEV in comparison to the G-SPCA. This aligns with the findings of Zou et al. (2006), who established a consistently lower PEV of 2.5% for SPCA when compared to the *simple-thresholded* Sparse Principal Component Analysis.

The implementation of SPCA in this study utilized the sklearn elastic net solver (Pedregosa et al., 2011). In order to obtain a percentage of non-zero loadings exceeding 40%, it was necessary to use very low values of regularization, resulting in $\alpha \leq 0.0001$, $\lambda \leq 0.0000045$ and $\lambda_1 \leq 0.00001$. However, the documentation of sklearn mentions that α should not be equal to 0 for numerical stability reasons. It seems that very low values of alpha should also be avoided, as the numerical stability of the elastic net regression was compromised for such low values of α.

One of the reasons to perform SPCA is because it might make complex machine learning models more explainable as a subset of the data is used. However, from the PEV plots, we find that the SPCA algorithm as implemented from the description of (Zou et al., 2006) does not provide a much more explainable model. Although SPCA shrinks loadings to zero, very high regularization is needed to set a row of loadings to 0, eliminating a feature from the principal components. Therefore, added explainability is only achieved when using very strong regularization.

5.3 Classification performance

No clear winner emerged between SPCA and G-SPCA in terms of classification performance. PCA however did outperform or tie with SPCA or G-SPCA on 3 out of 4 datasets. LR outperforms LGBM on three out of four datasets when transforming the data with PCA or SPCA. When the data is transformed with G-SPCA, LGBM outperforms LR on all datasets for both metrics. Classification performance might have proven more insightful when ran on more datasets. Out of 22 similar datasets, only these four were feasible as they had no more than 2900 features. Furthermore, although results seemed to vary with varying regularization and $l1$-ratio during early testing, a search over these parameters proved infeasible. Although an accuracy of around 90% was achieved for three of the four datasets and roughly 70% for the data from Gravier et al. (2010), results proved very sensitive to hyperparameter changes as sample sizes were limited.

5.4 Concluding remarks

The objective of this research was to examine the effects of various (Sparse) Principal Component Analysis methods on the performance and interpretability of classification models when applied to gene expression data. The findings indicate that there is little differentiation in terms of classification performance among the SPCA variants. Furthermore, the use of SPCA does not significantly enhance the interpretability of the results compared to G-SPCA, which often achieves a higher proportion of explained variance with a similar level of regularization. Additionally, SPCA is significantly slower in runtime compared to G-SPCA and PCA, thereby presenting challenges in hyperparameter optimization processes. Ultimately, when prioritizing classification performance and dimension reduction, PCA is recommended, while G-SPCA is favoured over SPCA when sparse loadings are necessary due to its faster runtime.

5.5 Future research

In this research we were limited to datasets with a maximum of around 2900 features due to limited computational power. It would be interesting to compare results performing the methods on datasets with a larger number of features. Additionally, it would be interesting to measure performance of alternative dimensionality reduction methods on gene expression data. For example, Xiang et al. (2021) suggest that *t*-distributed stochastic neighbour embedding (t-SNE) and uniform manifold approximation and projection (UMAP) might also be suitable dimensionality reduction methods for gene expression data, performing well in accuracy, computational cost and stability.

References

Aguirre-Gamboa, R., Gomez-Rueda, H., Martınez-Ledesma, E., Martınez-Torteya, A., Chacolla-Huaringa, R., Rodriguez-Barrientos, A., Tamez-Pena, J. G., & Trevino, V. (2013). Survexpress: An online biomarker validation tool and database for cancer gene expression data using survival analysis. *PloS one*, *8*(9), e74250.

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2623–2631.

Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., & Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, *96*(12), 6745–6750.

Anderson, J., & Blair, V. (1982). Penalized maximum likelihood estimation in logistic regression and discrimination. *Biometrika*, *69*(1), 123–136.

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, *24*.

Blandin Knight, S., Crosbie, P. A., Balata, H., Chudziak, J., Hussell, T., & Dive, C. (2017). Progress and prospects of early detection in lung cancer. *Open biology*, *7*(9), 170070.

Camacho, J., Smilde, A. K., Saccenti, E., & Westerhuis, J. A. (2020). All sparse pca models are wrong, but some are useful. part i: Computation of scores, residuals and explained variance. *Chemometrics and Intelligent Laboratory Systems*, *196*, 103907.

Chen, Y., Li, Y., Narayan, R., Subramanian, A., & Xie, X. (2016). Gene expression inference with deep learning. *Bioinformatics*, *32*(12), 1832–1839.

Debela, D. T., Muzazu, S. G., Heraro, K. D., Ndalama, M. T., Mesele, B. W., Haile, D. C., Kitui, S. K., & Manyazewal, T. (2021). New approaches and procedures for cancer treatment: Current perspectives. *SAGE open medicine*, *9*, 20503121211034366.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, *38*(4), 367–378.

Gravier, Pierron, G., Vincent-Salomon, A., gruel, N., Raynal, V., Savignoni, A., De Rycke, Y., Pierga, J.-Y., Lucchesi, C., Reyal, F., Fourquet, A., Roman-Roman, S., Radvanyi, F., Sastre-Garau, X., Asselain, B., & Delattre, O. (2010). A prognostic DNA signature for T1T2 node-negative breast cancer patients. *Genes, Chromosomes and Cancer*, *49*(12), 1125–1125.

Guerra-Urzola, R., Van Deun, K., Vera, J. C., & Sijtsma, K. (2021). A guide for sparse pca: Model comparison and applications. *psychometrika*, *86*(4), 893–919.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, *24*(6), 417.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

Khan, J., Wei, J. S., Ringnér, M., Saal, L. H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C. R., Peterson, C., & Meltzer, P. S. (2001). Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, *7*(6), 673–679.

Lehmann, B. D., & Pietenpol, J. A. (2014). Identification and use of biomarkers in treatment strategies for triple-negative breast cancer subtypes. *The Journal of pathology*, *232*(2), 142–150.

Li, L., Jamieson, K., Rostamizadeh, A., Gonina, E., Hardt, M., Recht, B., & Talwalkar, A. (2018). Massively parallel hyperparameter tuning. *arXiv preprint arXiv:1810.05934*, *5*.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, *2*(11), 559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Sørlie, T., Perou, C. M., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., Hastie, T., Eisen, M. B., van de Rijn, M., Jeffrey, S. S., Thorsen, T., Quist, H., Matese, J. C., Brown, P. O., Botstein, D., Lønning, P. E., & Børresen-Dale, A.-L. (2001). Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, *98*, 10869–10874.

Torre, L. A., Siegel, R. L., Ward, E. M., & Jemal, A. (2016). Global cancer incidence and mortality rates and trends—an updateglobal cancer rates and trends—an update. *Cancer epidemiology, biomarkers & prevention*, *25*(1), 16–27.

Van Der Maaten, L., Postma, E., Van den Herik, J., et al. (2009). Dimensionality reduction: A comparative. *J Mach Learn Res*, *10*(66-71), 13.

Wold, S., Ruhe, A., Wold, H., & Dunn, W., Iii. (1984). The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses. *SIAM Journal on Scientific and Statistical Computing*, *5*(3), 735–743.

Xiang, R., Wang, W., Yang, L., Wang, S., Xu, C., & Chen, X. (2021). A comparison for dimensionality reduction methods of single-cell rna-seq data. *Frontiers in genetics*, *12*, 646936.

Yuan, F., Lu, L., & Zou, Q. (2020). Analysis of gene expression profiles of lung cancer subtypes with machine learning algorithms. *Biochimica et Biophysica Acta (BBA)-Molecular Basis of Disease*, *1866*(8), 165822.

Zhang, Y., & Ghaoui, L. (2011). Large-scale sparse principal component analysis with application to text data. *Advances in Neural Information Processing Systems*, *24*.

Zou, H., Hastie, T., & Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, *15*(2), 265–286.