UiO : Department of Technology Systems
University of Oslo

University of Oslo

FYS-STK

Using PCA for dimensionality reduction

Authors:
Øien B. Lasse

Supervisor:
Prof. Morten Hjorth-Jensen

December 18, 2022

Abstract

There always needs to be more data to fully model and predict a problem, or should it be stated there is never enough unique data? This report has done a shallow dive into PCA and compared the internal results for a strongly correlated dataset and one for a dataset with a weak correlation. When features of a dataset are strongly correlated, the prediction score is related to the number of principal components used, as well, as fewer principal components are needed to describe the data. For a dataset with weak correlation, strong results were not found to suggest using PCA to describe the data. For the strongly correlated dataset, the comparison was made with a self-written neural network for classification, whereas the weakly correlated data was tested on sickit-learns MLPClassifier The shortcomings of the method used in the analysis include analyzing only two test sets and utilizing only one classifying algorithm. In contrast, the test runs on different algorithms would ensure an unbiased categorization of PCA.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Until now, there has been a significant focus on the core subjects of machine learning, and the author would like to dedicate this report to more applied data analysis. As mentioned in project 2, a technique called PCA is used for dimensionality reduction. This report will utilize and compare a PCA analysis of the Wisconsin breast cancer data and the classification accuracy of the neural network in project 2. A second data set with fewer entries and features will also be used to compare whether PCA is suitable for small data with less correlation. Both data sets will be run two times, one with enough PC(principal components) to describe 90% of the variance and a smaller set to investigate accuracy improvement with variance description of strongly correlated vs. not-so-strongly correlated datasets.

After a PCA analysis, the WBCD (Wisconsin breast cancer data) was run through the neural network described and created for project two[1]; this is to look further into the changes for optimal activation functions, neurons and hidden layers etc. From the second project, it was observed that sigmoid activation had a dominant role, whereas the argued Relu and Leaky Relu underperformed. This comparison provides insight into computation time vs accuracy payoff, as Leaky Relu and Relu are said to have less computational requirements than Sigmoid. The wine recognition dataset was run through the MLP classifier by sickit-learn. This is an easy yet effective benchmark check for using PCA on small datasets. The use of these external solutions is needed as PCA in itself may produce a good separation of classes; this report focuses mainly on PCA as a dimensionality reduction method. But it can also be used for predictive regressions (PCR)

The motivation for doing PCA is to do something more applied, and from the authors' field, it is one of many considered models in classifying consumer patterns. The interest of the analysis is also to compare the difference in the value of conducting a PCA based on data entries and features. This is to get a feel for when particular analysis techniques should be performed and when not to use the method.

So what is PCA? Well, PCA is a form of reducing dimensionality in a problem. The superficial way it completes this is to split the data based on specific values and then reduce many correlated features into the principal component. This is done for large datasets because it takes low computational effort and reduces the dimensionality problem data you want to feed into a neural network, support vector

machine, etc. The need and usage have proven valuable in data analysis, especially in the pharmaceutical field. With fewer features, the data is more easily processed.

Chapter 2

THEORY

> It is not enough to have a good mind; the main thing is to use it well.
>
> Rene Descartes

While Descartes's quote referred to the human mind, it applies to the world of machine learning and the use of clever techniques to fulfill a goal. One of these innovative techniques is PCA, which lowers the dimensions of the problem and separates powerfully correlated factors, which makes neural networks or support vector machines' jobs easier and may produce better results. This section will go through the mathematics and concept related to correlation and PCA. Most of this section will be theory obtained from the lecture notes[2].

2.1 Correlation and Co variance

Since PCA is linked to correlation and covariance, a brief introduction/reminder will be given in this section.

Given the existence of two vectors $\hat{x}$ and $\hat{y}$ with n elements each. One can define the covariance[1] as follows.

$$\boldsymbol{Cov}(\boldsymbol{x}, \boldsymbol{y}) = \mathbf{E}[(\boldsymbol{x} - \mathbf{E}(\boldsymbol{x})) \cdot (\boldsymbol{y} - \mathbf{E}(\boldsymbol{y}))] \tag{2.1}$$

Expecting the value to be its mean, the covariance can be rewritten as

$$\boldsymbol{Cov}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n} \sum_{i=0}^{n-1} (x_i - \overline{x})(y_i - \overline{y}) \tag{2.2}$$

Thus leading to the covariance matrix as.

$$\boldsymbol{Cov}(\boldsymbol{x}, \boldsymbol{y}) = \begin{bmatrix} \text{cov}[\boldsymbol{x}, \boldsymbol{x}] & \text{cov}[\boldsymbol{x}, \boldsymbol{y}] \\ \text{cov}[\boldsymbol{y}, \boldsymbol{x}] & \text{cov}[\boldsymbol{y}, \boldsymbol{y}] \end{bmatrix} \tag{2.3}$$

Recognizing that $cov[x, x] = \text{var}[x]$. The covariance matrix can be rewritten as

[1] Covariance is defined as : For two real-valued random variables, the covariance is the expected value of the product of their individual expected value deviation from their actual value

$$\boldsymbol{Cov}[\boldsymbol{x}, \boldsymbol{y}] = \begin{bmatrix} \text{var}[\boldsymbol{x}] & \text{cov}[\boldsymbol{x}, \boldsymbol{y}] \\ \text{cov}[\boldsymbol{x}, \boldsymbol{y}] & \text{var}[\boldsymbol{y}] \end{bmatrix} \tag{2.4}$$

Since the covariance has no restriction between zero and infinity, significant values may occur, giving rise to the loss of numerical precision. It is common to scale and define the correlation matrix to deal with such a problem. It was first defining the correlation function.

$$\text{corr}[\boldsymbol{x}, \boldsymbol{y}] = \frac{\text{cov}[\boldsymbol{x}, \boldsymbol{y}]}{\sqrt{\text{var}[\boldsymbol{x}]\text{var}[\boldsymbol{y}]}} \tag{2.5}$$

Following the equation will lead to a matrix with ones along the main diagonal and correlations on the off-diagonal.

$$\boldsymbol{K}[\boldsymbol{x}, \boldsymbol{y}] = \begin{bmatrix} 1 & \text{corr}[\boldsymbol{x}, \boldsymbol{y}] \\ \text{corr}[\boldsymbol{y}, \boldsymbol{x}] & 1 \end{bmatrix} \tag{2.6}$$

With the correlation matrix defined, a valuable way to describe the PCA is now.

2.2 PCA Theorem

Since there are many ways to define PCA, and there are some theorems that do very well of describing them with rigorous math. This section will give a more intuitive way of describing PCA in detail and include some math for two reasons. One to not kill the reader's enthusiasm; two, this paper was written by an engineer[2]. More theorems and proofs can be found in [3, 4].

To better grasp the PCA, the author will first introduce a broader intuitive way with a simple 2D case that is our data is described by two features. Imagine each feature represented as a line perpendicular to one another, meaning they make up a plane. In this plane, some data points represent the samples. The main idea now from the PCA is to draw a straight line through the plane's origin to minimize the length from the line to each data point. In other words, it is the line that minimizes the sum of squared distances from the line to each point or maximizes the sum of squared distances from the origin to the point when projected on the line. This line is now the first new principal component axis. The principal component can be described by a linear combination of these two features, and the second principal component is a line that is perpendicular to the first line. These two new perpendicular lines create a hyperplane; this plane is made up of the principal component PC1, which contains both of the features, as well as PC 2, which is a line perpendicular to it.

The sum of squared distances for the best fit line is the eigenvalue for the first PC (principal component), and the square root is the singular value for the principal component. From project one there was introduced the theorem of SVD (Singular value decomposition)

[2]In Meme culture engineers are known by mathematicians and physicists for their easy way out approach and extended assumptions. Also, integrals and theorems that can't be described in the lookup table are rarely considered applied and do not bother engineers

The relation to SVD can be as quickly described as follows. Imagine the feature matrix $\boldsymbol{X}$ with dimensions nxp and assume it is centered and scaled. Then the covariance matrix for pxp is given as:

$$\boldsymbol{Cov}[\boldsymbol{x}] = \frac{1}{n}\boldsymbol{X}^T\boldsymbol{X}$$

And since it is a symmetrical matrix, it can be diagonalized.

$$\boldsymbol{Cov} = \boldsymbol{VLV^T}$$

Where $\boldsymbol{V}$ is the eigenvector matrix and $\boldsymbol{L}$ is the diagonal eigenvalue matrix with each element λ_i in decreasing order alongside the main diagonal. The principal component axis (the line described earlier with points projected on) is given by the product $\boldsymbol{XV}$ such that the j-th column of the product provides the j-th principal component and the i-th datapoints coordinate in the pc space is given by the i-th row in the product[4, 5, 6].

The SVD of $\boldsymbol{X} = \boldsymbol{USV^T}$ where $\boldsymbol{U}$ is a unitary matrix and $\boldsymbol{S}$ is the diagonal matrix with the singular values[7]. It can be used to rewrite the covariance matrix.

$$\boldsymbol{C} = \frac{1}{n} \cdot \boldsymbol{VSU^T} \cdot \boldsymbol{USV^T} = \boldsymbol{V}\frac{\boldsymbol{S^2}}{n}\boldsymbol{V^T} \tag{2.7}$$

This relationship serves to show that the eigenvalues of the covariance matrix are related to the singular values of the feature matrix such that $\lambda_i = \frac{s_i^2}{n}$ Applying the relationship described previously in this section results in the principal components can be written as.

$$\boldsymbol{XV} = \boldsymbol{US} \tag{2.8}$$

Note that this is just one of many ways to describe the principal component analysis and derive the principal components. More can be found in the lecture notes [2], and the books [3, 4].

Each principal component accounts for some given % of the variance within the dataset. The key for each principal component is to maximize the variance; this is because variance is a measure of the spread of data. Thus maximizing it will give each principal component vector uniqueness. The more unique the components are, the more spread out they are, and thus, easier to cluster data that are correlated together.

The Matrix now conducted in equation (2.8) can be fed into a support vector machine, random forest algorithm or neural network. This report utilizes a standard feed-forward neural network with backpropagation and uses cross entropy as a cost score. The neural network is described more in detail in project two [1]. The other network that is used is the MLP classifier by sickit learn. This neural network allows for different gradient descent methods and adaptable learning rates.

Chapter 3

METHOD

This section will go through the method used, as always a brief introduction of the data used. And the main focus of this chapter is on how the PCA is conducted on breast cancer data. Since the breast cancer data was introduced in project 2, the data presented in this method chapter will be the wine recognition dataset.

3.1 Wine recognition dataset

The wine recognition dataset is a classification dataset consisting of 3 classes, 13 features, and 178 entries. Previously this data has been used to compare supervised learning techniques such as RDA, LDA, and QDA[8].

The features are as follows:

- Alcohol
- Malic acid
- Ash
- Alcalinity of ash
- Magnesium
- Total phenols
- Flavanoids
- Nonflavanoid phenols
- Proanthocyanins
- Color intensity
- Hue
- OD280/OD315 of diluted wines
- Proline

Class 0, class 1, and class 2, where samples pr class are in the separate order 59,71, and 48, which means 40% accuracy by guessing on class 2. Compared to the Wisconsin breast cancer data, guessing benign would result in 63% accuracy.

Looking at how many of the features were strongly correlated with other features of the dataset, it is evident that the Wine data is harder to reduce dimensionality, for this report, the correlation between two features was set to 0.61.

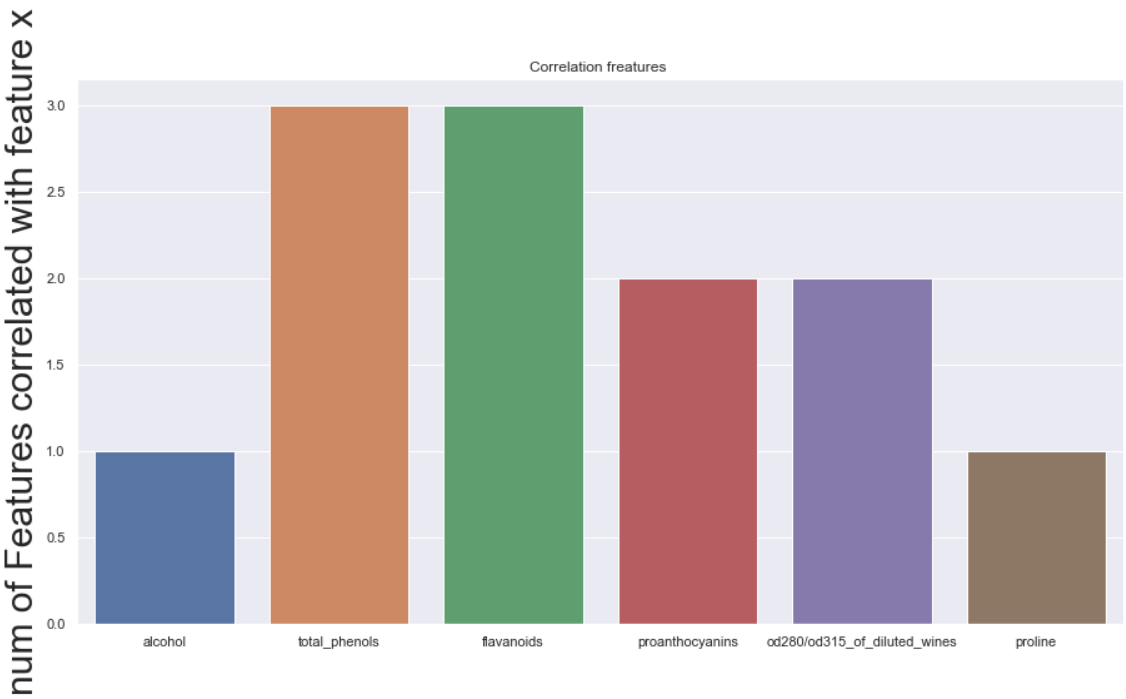


Figure 3.1: Number of strongly correlated features for the wine data.

Figures (3.1 & 3.2) can be interpreted as follows: The x-axis is the features that have a strong correlation with other features in the datasets. If the correlation matrix gave a score above 0.61 between one feature and another, it was added as one strong correlation for both features. By looking at the figure (3.1) and comparing it with the number of features in the dataset, it is evident that this dataset has a weak correlation. Looking at WBCD in figure (3.2) where it is observed that some of the features are strongly correlated with over half of the total number of features.

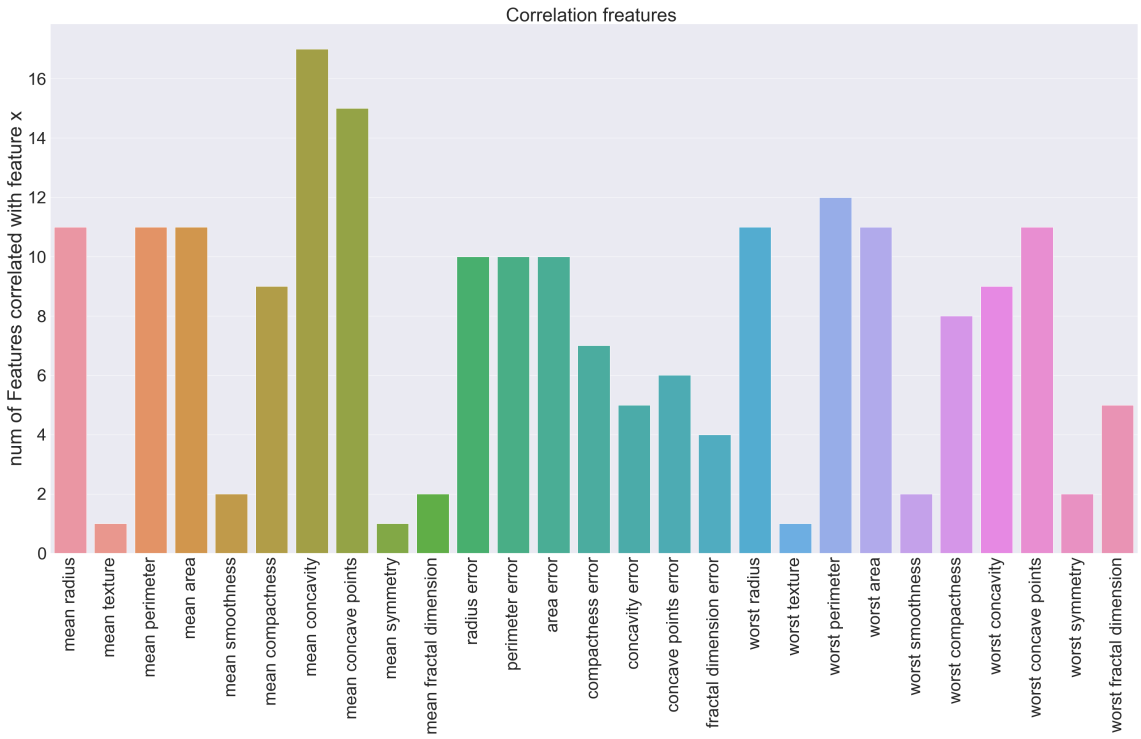


Figure 3.2: Correlation of features for WBCD

3.2 PCA-Method

This section will contain a step-by-step guide on PCA. The results from running the reduced dimensionality through the neural network for the Wisconsin breast cancer data can be found in the result section, together with the classification test of PCA vs. non-PCA of the wine recognition dataset utilizing the MLP classifier. This section mainly focuses on the step-by-step guide on PCA analysis done on the Wisconsin breast cancer dataset.

Firstly, as mentioned in the theory chapter, scaling the feature matrix utilizing sickit-learn standard scaler. Then split the data into train and test sets (for illustrative purposes, this is not done in this section). The next step is selecting how many principal components to describe the data, as described in the theory chapter, the maximum variance with little redundancy. This can be viewed in the scree plot.

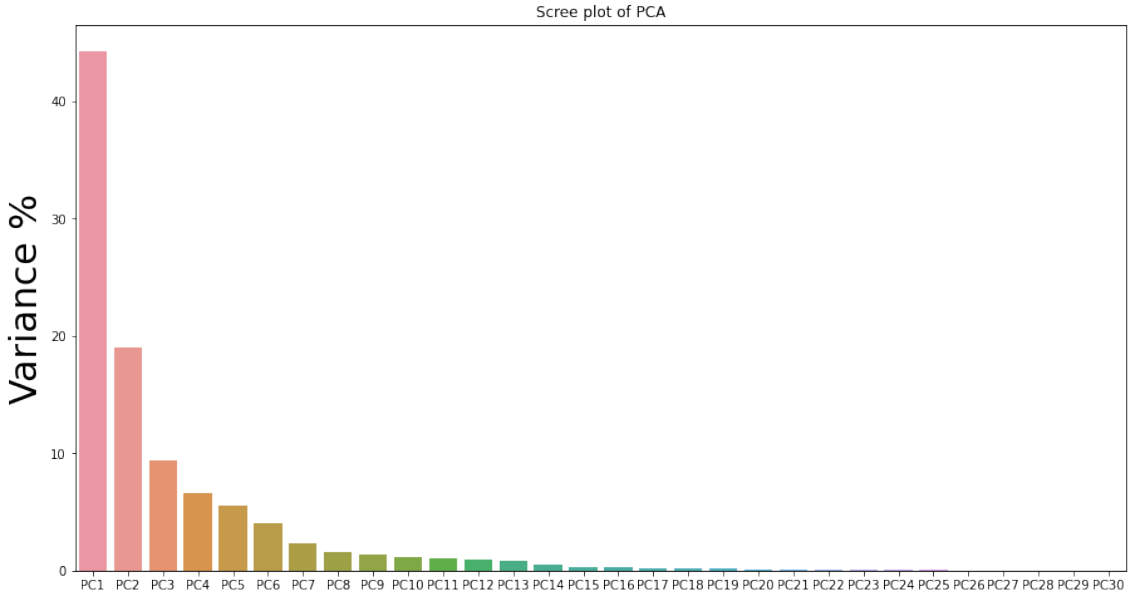


Figure 3.3: Scree plot of WBCD

From the plot, it is observed that PC1 accounts for 44.3% of the variance in the data; based on this plot, 3 Principal components were selected. For further analysis, a standard method is to have as many principal components where the cumulative sum of the principal component's variance reaches 95% or more. The cumulative sum of these principal components is 74%. Note using seven principal components would result in 90% of the variance being covered; thus, one run with three principal components and seven were conducted for the result section on breast cancer data, and five and seven for the wine recognition dataset.

The next step is to visualize how well the principal component divided the classes where red is malignant tumors and blue is benign.

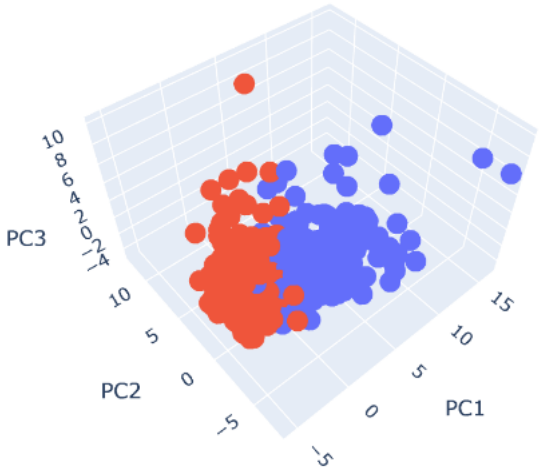
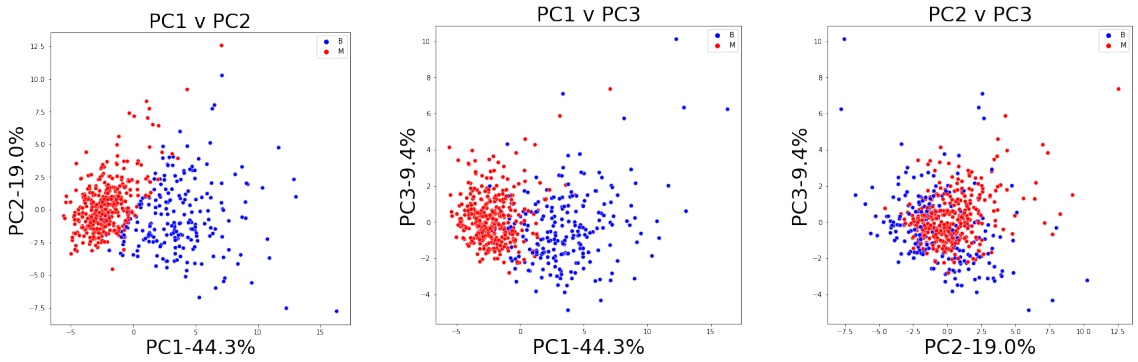


Figure 3.4: 3d-plot of PCA split

From the figure, it is clear that most malignant and benign cluster together with

little spread along the PC3 axis. Observing the 2d plots expressing the data through PC1 and PC2 gives a good clustering of the malignant with some spread along the PC2 axis.



(a) PC1 v PC2 (b) PC1 v PC3 (c) PC2 v PC3

Figure 3.5: 2d plot of the PCA axes

After observing how accurate the PCA clustering is, the standard question is to ask what features for each principal component were significant drivers for splitting the data. This is commonly referred to as loading scores.

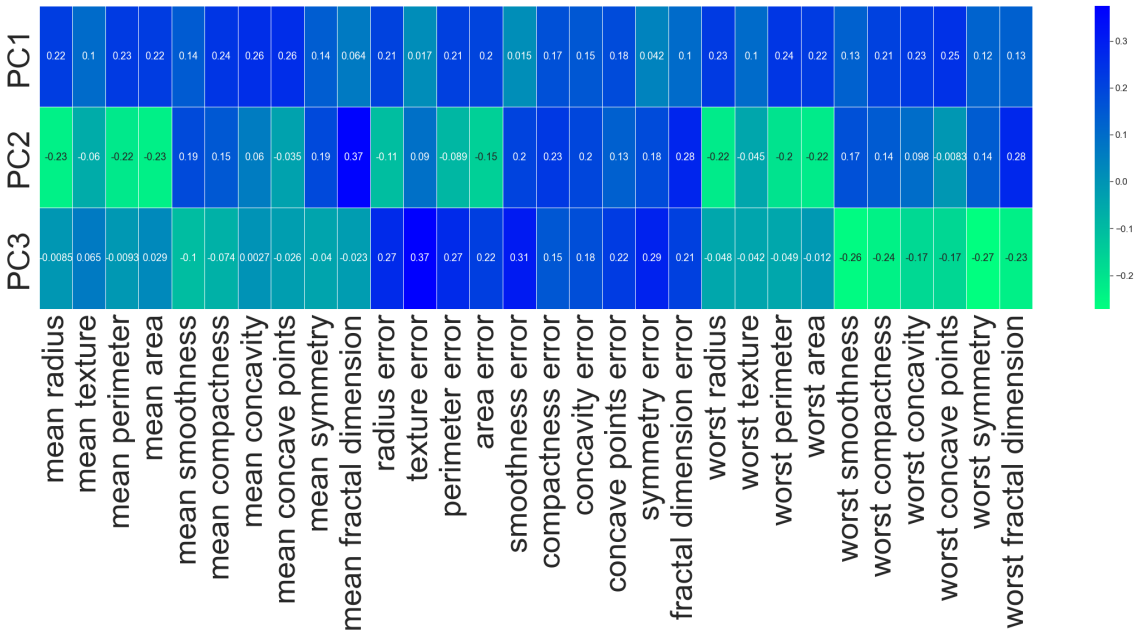


Figure 3.6: Loading scores of PC1,2,and 3

The loading scores for PC1 are more stable than for PC2 and three, as observed from the color scheme. This is interpreted as more features are necessary to split the classes in PC 1 than in PC 3.

Lastly the matrix containing the PC vectors as columns are fed into the neural network and, thus, contains fewer dimensions, three in this case, than the thirty features fed into the neural network in project 2.

To summarize the proceedure:

- split into train and test set then scale.
- Define how many principal components
- feed the reduced dimensionality matrix into a neural network for comparison.
- Observe if it is a good fit.
- Check the loading scores.

3.3 Analysis

After the PCA is conducted and the matrix containing the PC vectors is created, the next step is to do the analysis. The analysis consits of checking the number of principal components against training convergence and prediction accuracy.

As mentioned, the WBCD will be run on the self-written neural network to compare changes in optimal value using PCA vs non-PCA. The analysis was conducted in the following manner. Import the PCs and split them into train and test sets, then train the network in a similar fashion as in project two [1]. Locate the activation functions with the highest average accuracy over 100 epochs, then plot these for comparison with the results in project two.

For the wine recognition data, the PC components are fed into the MLPclassifier by sickit learn, utilizing the adam optimizer. The accuracy score is computed over 250 epochs for training and test data. The analysis then compares the PC data training convergence vs the entire dataset. Then the comparison and judgment will be made on training convergence and test accuracy.

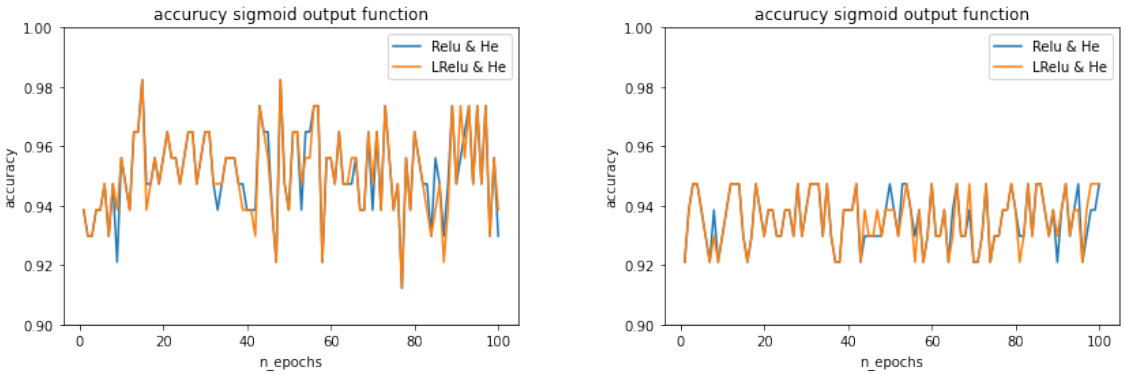
Chapter 4

RESULT/DISCUSSION

This section will include all the results and a brief discussion of them. The discussion part is to discuss whether or not the results can answer the questions and to what extent they succeed or fail. First, there will be a comparison between the neural net results from project two and the PCA from this project. Secondly, the PCA will compare with a smaller dataset for the wine recognition data.

4.1 Breast cancer neural net

In this section, the neural net created by the author for project two will be tested on the same data, just utilizing PCA. The first noticeable change from project 2 is that the activation function for optimal solution changed compared to project two and the number of principal components used in the analysis.



(a) Sigmoid output 7 PC 90% Variance

(b) Sigmoid output 3PC 74% Variance

Figure 4.1: Sigmoid in output layer PCA

The figure shows the two best activation functions with corresponding weight initialization for both PCA run through the neural net. Using seven principal components corresponding to seven features for the neural network leads to higher accuracy, and the mean accuracy corresponds to the best mean accuracy without PCA.

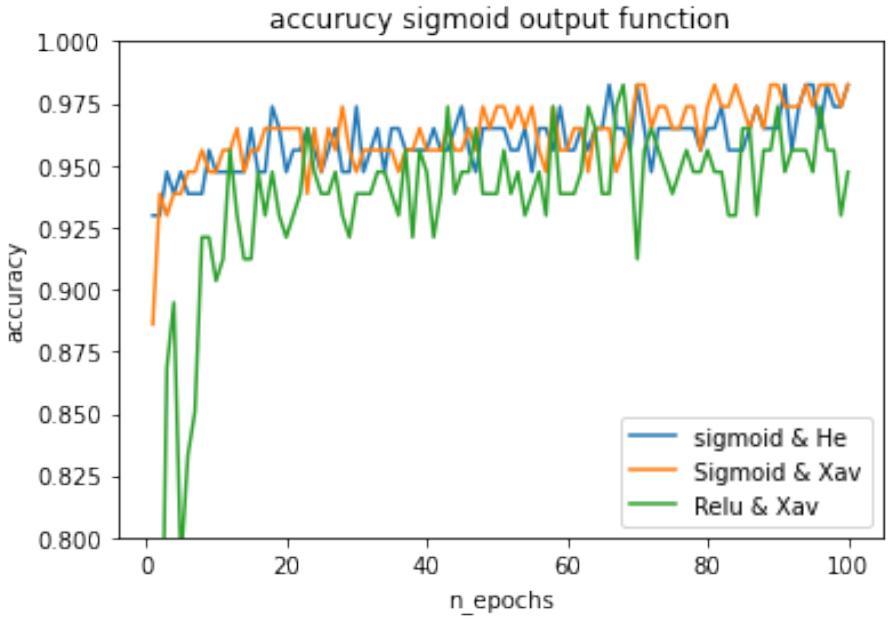


Figure 4.2: Neural network with no PCA Sigmoid outputlayer

In contrast with the PCA run, the activation functions are more sigmoid-favored when using all the features as id. However, Relu and leaky Relu were spotted at the highest accuracy peaks, but before any convergence was observed.

Utilizing the sigmoid in the output layer, there are some benefits to using the PCA. This is because the Relu and Leaky Relu performs almost as well as sigmoid without PCA. Again, it needs to be checked towards 250 epochs which would give a more apparent convergence. The PCA with more variance expression experiences more turbulent progression in the epochs. This may hint at slower convergence criteria and differs from what one hopes to see when reducing dimensionality. The tradeoff lies not in the accuracy of this dataset but may have been in the number of hidden layers/Neurons and convergence criteria. For this dataset, this needed to be investigated thoroughly to give a solid answer. Using the Sigmoid in the output layer did not prove or disprove any significant increase in accuracy or convergence. Still, the reduction of dimensionality with less than half of the features reached equivalent scores as no dimensionality reduction. The code and analysis can be viewed in the GitHub

From project two, using tanh in the output layer matched the sigmoid case, and using different output layer functions was unnecessary. That is, the results were quite similar, with still the sigmoid as activation function being the highest accuracy.

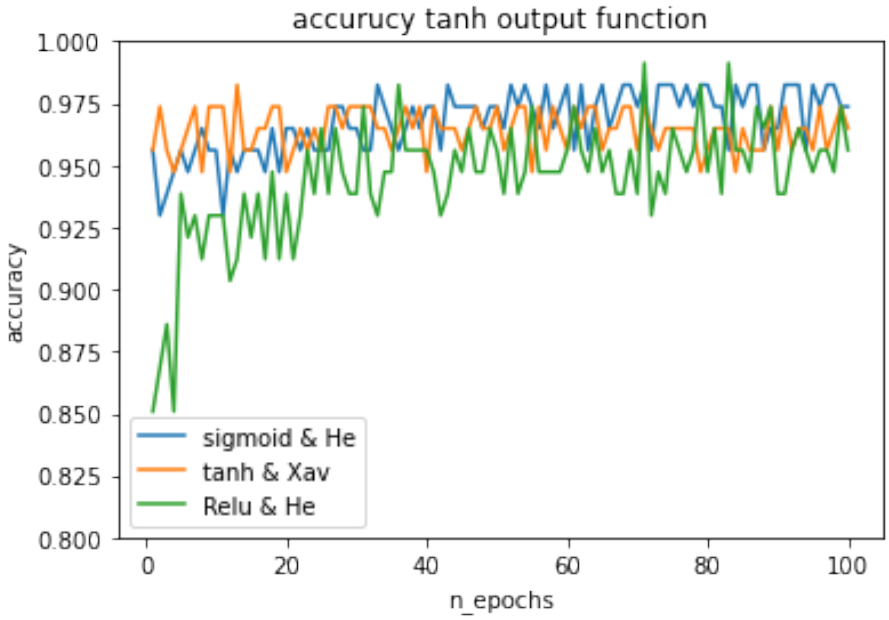
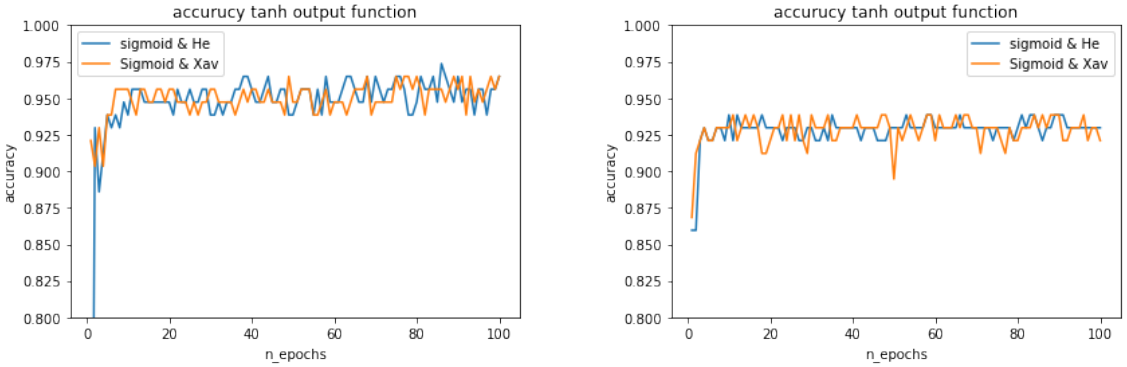


Figure 4.3: project two results tanh

While using the PCA, the results are as follows.



(a) tanh output 7 PC 90% Variance

(b) tanh output 3PC 74% Variance

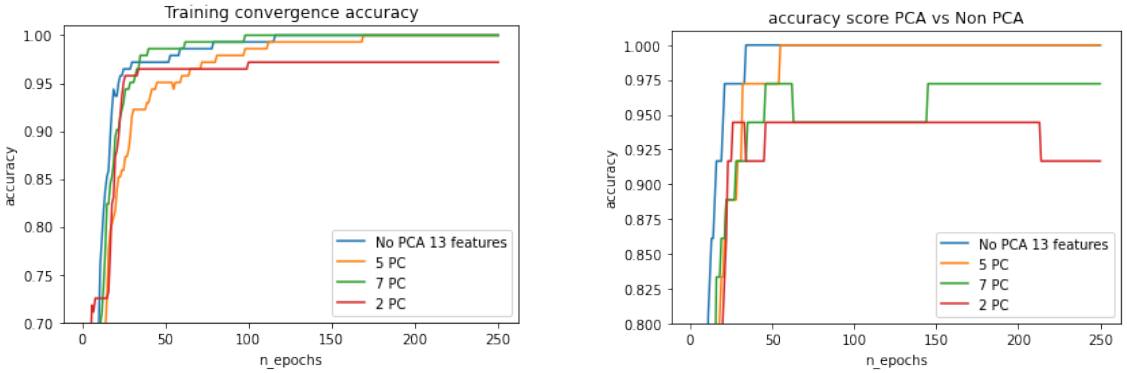
Figure 4.4: tanh in output layer PCA

The figures show thaconverges fastergence when using the PCA and tanh in the output layer. But lacks in the accuracy score. The lack of accuracy would be interesting to see convergence and score with 90% of the variation handled by PCA; the atlas comparing PCA to PCA agrees with a dataset as WBC (Wisconsin breast cancer) that describes the most variance seems to be correlated with an accuracy score.

4.2 Wine Recognition dataset

For the Wine recognition dataset, the method described in the method chapter can be viewed on GitHub. One significant difference is using the MLP classifier instead of the self-written neural net. This is to save time and focus more on the appliance of PCA. The test was run on seven principal components for the 90% and

five principal components for 80% of variance; this was compared by utilizing all 13 features. Lastly, two principal components were added to help give a conclusion.



(a) Training converegence

(b) Prediction scores

Figure 4.5: Wine recognition dataset

From this dataset, the author is surprised that higher variance coverage did not contribute to better accuracy but resulted in better training convergence. Another observation is that the lower the variance description, the slower the training convergence, but there is a lack of evidence for the number of principal components and test accuracy. Whereas WBCD test scores gave higher accuracy based on the number of principal components and variance described, the wine data seems to differ. This may be because the covariance and correlation between features and targets in this dataset are less correlated than the features in WBCD.

One reason for 80% variance coverage (PC 5) is better than 7, might be because of the the extra principal components do not contribute too much more variance coverage pr PC. This may serve an adverse effect andor confusion for the neural network. For further investigation, more PC will be run and added.

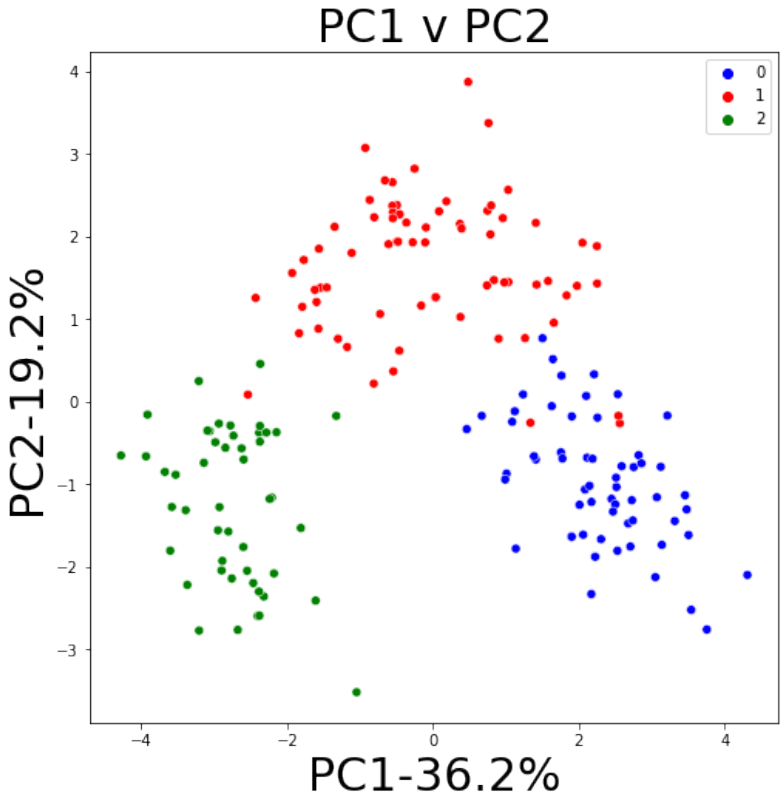
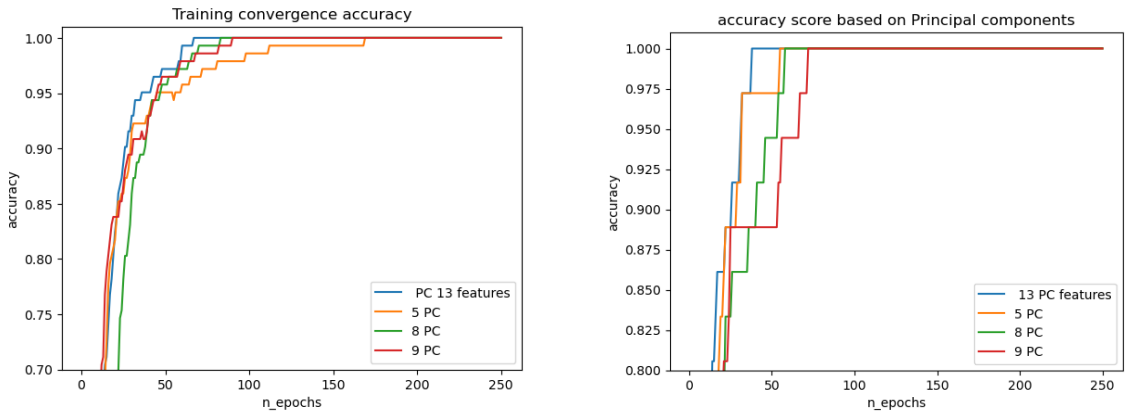


Figure 4.6: PC1 V PC2 Wine data

In the figure displaying the clustering of PC1 and PC2, the PCA does an excellent job separating the classes, and by using 80% of the variance, the accuracy of 100% was reached. However, it provides little argumentation since training convergence and score were better/equal without using PCA. A fault in this analysis is the lack of cross-validation. As most PCA is used in contribution with random forests or SVM, this report focused more on the value of PCA using standard supervised learning methods. Although, as the author hypothesized, the results of PCA matched the breast cancer data better than the wine data. This led to the intuition that the low correlation in datasets, the more features are needed to describe the data.

As an additional test in the face of these results, there was a setup to run with the number of principal components varying from 5 to 10 and 13. This was done to help clarify whether or not the number of principal components had an impact.



(a) Training converegence

(b) Prediction scores

Figure 4.7: Evaluating the number of principal components impact on training and test scores

From the figure, it seems that increasing the number of principal components and thus having more significant variance descriptions have a positive effect on training convergence. When analyzing the test scores, it does not seem to be a clear line between the number of PCs involved and how good predictions can be made. This also invalidates previous arguments that extra PCs give adverse effects and confusion. Simular findings were found when we

Chapter 5

CONCLUSION

After a discussion on PCA, the author sees the merits, although solid evidence for the use of PCA was not found. The report's scope may have focused on a narrow aspect when comparing PCA with non-PCA predictions. But to conclude on Rene Descartes's quote given in the theory chapter. PCA should mainly be used on datasets with more features when several features are strongly correlated; that is, use the mind well. As the WBCD gave similar to better results for sigmoid in the output layer and identical to the worse outcome for tanh, the upside is that the dimensionality reduction favored the use of Relu and Leaky Relu, which was argued as less computational heavy in project two.

One observation made clear during the analysis was that the increased variance description through PCA gives better results when the dataset is strongly correlated. The variance description had little to no impact for not strongly correlated features. However, the more variance that is described resulted in faster training convergence. No symmetry or a strong link was observed between the number of principal components and test accuracy for weak correlation datasets. However, there was an improvement in prediction scores related to the number of principal components when observing strongly correlated datasets. Another vital observation was that more correlated data, more variance description and fewer PC axes were needed for covering 90% variance.

For weakly correlated datasets, the use of PCA for dimensionality reduction gives benefits to the training convergence but does seem to be random when it comes to predicting. Although the it seems that similar or evenly good results can be matched, future work will investigate more models and more datasets to map out when PCA can be valid. Furthermore, by looking at the PCA plots for simple new addition cases and not bulk, placing new entries in a category seems good enough by just a PCA analysis.

Bibliography

[1] Lasse B Øien. *Project 2*. URL: https://github.com/lasseboi/Fys-STK/blob/main/Project%202/Report/Project_2_Report.pdf. (accessed: 02.12.2022).

[2] Morten Hjortjh-Jensen. *PCA and clustering*. URL: https://github.com/CompPhysics/MachineLearning/blob/master/doc/pub/week47/ipynb/week47.ipynbl. (accessed: 02.12.2022).

[3] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[4] René Vidal, Yi Ma, and S.S. Sastry. *Generalized Principal Component Analysis*. Vol. 40. Interdisciplinary Applied Mathematics. New York, NY: Springer New York, 2016. ISBN: 978-0-387-87810-2 978-0-387-87811-9. DOI: 10.1007/978-0-387-87811-9. URL: http://link.springer.com/10.1007/978-0-387-87811-9 (visited on 12/03/2022).

[5] Rene Vidal, Yi Ma, and Shankar Sastry. "Generalized principal component analysis (GPCA)". In: *IEEE transactions on pattern analysis and machine intelligence* 27.12 (2005), pp. 1945–1959.

[6] stack overflow user. *Relationship between SVD and PCA. How to use SVD to perform PCA?* URL: https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca. (accessed: 02.12.2022).

[7] Kirk Baker. "Singular value decomposition tutorial". In: *The Ohio State University* 24 (2005).

[8] Peter Hall, Yvonne Pittelkow, and Malay Ghosh. "Theoretical measures of relative performance of classifiers for high dimensional data with small sample sizes". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1 (2008), pp. 159–173.