A Comparative Study of Principal Component Analysis' Impact on Performance Metrics of Regression Models

A Fourth YEAR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
Math-451 (Math Seminar)

BY

Bishesh Kafle (Roll No.-09)

Anisha Lamsal (Roll No.-11)

Priyanka Panta (Roll No.-13)

Submitted To:
Prof. Dr. Kanhaiya Jha
Department of Mathematics
KATHMANDU UNIVERSITY
DHULIKHEL, NEPAL

February 8, 2024

CERTIFICATION

This project entitled "A Comparative Study of Principal Component Analysis Impact on Performance Metrics of Regression Models" is carried out under my supervision for the specified entire period satisfactorily, and in hereby certified as work done by the following students

1. Bishesh Kafle (09)
2. Anisha Lamsal(11)
3. Priyanka Panta (13)

in partial fulfillment of the requirements for the degree of B.Sc. in Computational Mathematics,Department of Mathematics,Kathmandu University,Dhulikhel,Nepal.

Dr. Kanhaiya Jha
Professor
Department of Mathematics,
School of Science,Kathmandu University,
Dhulikhel,Kavre,Nepal
Date:

APPROVED BY:
I hereby declare that the candidate qualifies to submit this report of the Math Seminar(MATH 451) to the Department of Mathematics.

Head of Department
Department of Mathematics
School of Science
Kathmandu University
Date:

ACKNOWLEDGMENTS

We would like to express our sincere gratitude towards our instructor and supervisor Prof. Dr. Kanhaiya Jha for providing immense support and invaluable guidance to undertaking this research endeavor. We are very grateful to him for imparting insightful knowledge and skills required in this project.The unwavering guidance, support, and expertise has been crucial in shaping the content of the report as well. Additionally, we would like to thank the entire Department of Mathematics for their support and encouragement throughout the project.

Moreover, we are indebted to our friends for motivating us throughout. Their contributions have been pivotal in the successful completion of this project.

Lastly, we would like to thank everyone who helped us directly and indirectly during the completion of our project work.

ABSTRACT

In today's data-driven landscape, making sense of vast datasets is a crucial challenge. Our project dives into the world of Principal Component Analysis (PCA), a powerful tool that helps make sense of large datasets.This project conducts a thorough exploration into the impact of PCA on regression models, those mathematical statistics predicting relationships between variables.PCA are also applied to real-world issues like financial forecasts and medical diagnoses.This study examines into how PCA influences performance metrics like Mean Squared Error (MSE), R-squared, and interpretability of regression models, while also analysing timing and computational efficiency.The approach involves collecting applicable datasets, handling missing values, and normalizing features. By comparing baseline regression models with those augmented by PCA, the study aims to focus on the performance improvement, insights into dimensionality reduction, and a clearer understanding of computational efficiency.

Keywords : PCA, Mean Square Error, dimensionality redution, R-squared.

CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Principal Component Analysis(PCA)

Large datasets are increasingly widespread in many disciplines. In order to interpret such datasets, methods are required to drastically reduce their dimensionality in an interpretable way, such that most of the information in the data is preserved. Many techniques have been developed for this purpose, but Principal Component Analysis (PCA) is one of the oldest and most widely used. PCA is a statistical technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss.[1]

PCA is a very flexible tool and allows analysis of datasets that may contain, for example, multicollinearity, missing values, categorical data, and imprecise measurements. The goal is to extract the important information from the data and to express this information as a set of summary indices called principal components[2].Statistically, PCA finds lines, planes and hyper-planes in the K-dimensional space that approximate the data as well as possible in the least squares sense. A line or plane that is the least squares approximation of a set of data points makes the variance of the coordinates on the line or plane as large as possible.

1.2 Background

Principal Component Analysis has a rich historical background that traces back to the early 20th century. PCA was invented in 1901 by Karl Pearson (LI, 1901), who formulated the analysis as finding "lines and planes of closest fit to systems of points in space."

PCA was briefly mentioned by Fisher and MacKenzie as more suitable than analysis of variance for the modeling of response data. Fisher and Mackenzie also outlined the Nonlinear Iterative Partial Least Squares (NIPALS) algorithm, later rediscovered by Wold. Hotelling further developed PCA to its present stage[3].

PCA is often regarded as the first true 'multivariate' statistical method. PCA's essential purpose is to understand the best fit plane through a system of points in multidimensional space, thereby distilling complex interactions among variables to essential unifying relationships represented by this lower dimensional plane (technically known as a 'hyperplane'). In this sense, PCA can be thought of as a multivariable form of the Pearson correlation, where the best-fit line has been replaced by a best fit, hyperplane. PCA is closely related to the technique of exploratory factor analysis (EFA) developed by Charles Spearman in 1904 for developing psychometric scales, and the two approaches are often used interchangeably.

Depending on the field of application, it is also named the discrete Karhunen–Loève transform (KLT) in signal processing, the Hotelling transform in multivariate quality control, proper orthogonal decomposition (POD) in mechanical engineering, singular value decomposition (SVD) of X (invented in the last quarter of the 19th century[11]), eigenvalue decomposition (EVD) of XTX in linear algebra, factor analysis, Eckart–Young theorem (Harman, 1960), or empirical orthogonal functions (EOF) in meteorological science (Lorenz, 1956), empirical eigenfunction decomposition (Sirovich, 1987), quasiharmonic modes (Brooks et al., 1988), spectral decomposition in noise and vibration, and empirical modal analysis in structural dynamics[3].

1.3 Objectives

The key objectives of our study are stated as follows:

1. To conduct a comparative analysis of linear and logistic regression models on high-dimensional data with and without PCA preprocessing.

2. Apply dimensionality-reduced models to real-world regression problems, such as financial forecasting or medical diagnosis, to assess their practical utility.

3. Acquire a deeper understanding of the specific domain or application area where regression modelling and dimensionality reduction are relevant.

4. Investigate the computational efficiency of regression models with PCA compared to those without PCA,aiming to identify time complexity and memory usage.

CHAPTER 2

Mathematical Notions

2.1 Linear Regression

Linear Regression is a statistical method of fitting first degree equation describing relation between dependent (or response) variable (y) and a single explanatory (or independent) variable (x) to given set of data. In simple regression following model equation is fitted to the available/observed set of paired data on (x, y) :

$$y = \alpha + \beta x$$

Multiple regression is an extension of simple regression in which more than one independent variables (or explanatory variables) affecting a dependent variable (or response variable) are considered in regression analysis. If are k – independent variables affecting the dependent variable , then model equation representing relationship between dependent and independent variables is expressed in following form-

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + ... + \theta_k x_k + \epsilon$$

2.1.1 Cost Function in Linear Regression

Cost function is the calculation of the error between predicted values and actual values, represented as a single real number.The difference between the cost function and loss function is as follows:

The cost function is the average error of n-samples in the data (for the whole training data) and the loss function is the error for individual data points (for one training example).

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x_i) - y]^2$$

where, x = Predicted Value, y = True Value

2.1.2 Gradient Descent in Linear Regression

Gradient Descent is an iterative optimization algorithm that tries to find the optimum value (Minimum/Maximum) of an objective function.The main aim of gradient descent is to find the best parameters of a model which gives the highest accuracy on training as well as testing datasets. In gradient descent, the gradient is a vector that points in the direction of the steepest increase of the function at a specific point. Moving in the opposite direction of the gradient allows the algorithm to gradually descend towards lower values of the function, and eventually reaching to the minimum of the function.

$$\theta_j = \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1)$$

where, α = Learning Rate

Now,

$$\frac{\delta}{\delta \theta} J_\theta = \frac{\delta}{\delta \theta} \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x_i) - y]^2 = \frac{1}{m} \sum_{i=1}^{m} [h_\theta(x_i) - y] \frac{\delta}{\delta \theta_j} (\theta_{x_i} - y) = \frac{1}{m} [h_\theta(x_i) - y] x_i$$

Therefore,

$$\theta_j = \theta_j - \frac{\alpha}{m} \sum_{i=1}^{m} [(h_\theta(x_i) - y) x_i]$$

2.2 Logistic Regression

In statistics, the logistic model (or logit model) is a statistical model that models the log-odds of an event as a linear combination of one or more independent variables. In regression analysis, logistic regression[1] (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination). The logistic regression model transforms the linear regression function continuous value output into categorical value output using a sigmoid function, which maps any real-valued set of independent variables input into a value between 0 and 1. This function is known as the logistic function.

The logistic function is of the form:

$$\sigma = \frac{1}{1 + e^{-}z^{(i)}}$$

where,

$$z^{(i)} = h_\theta = \theta_1 x_i + \theta_2$$

is the linear combination of input features and model parameters,
The predicted probability σ is the output of this sigmoid function,
e is the base of the natural logarithm, approximately equal to 2.718.

2.2.1 Log Loss in Linear Regression

Log loss is a classification evaluation metric that is used to compare different models which we build during the process of model development. It is considered one of the efficient metrics for evaluation purposes while dealing with the soft probabilities predicted by the model.

The log of corrected probabilities, in logistic regression, is obtained by taking the natural logarithm (base e) of the predicted probabilities.

$$logloss = \log \sigma = \log \frac{1}{1 + e^{-}z^{(i)}}$$

Then, the log loss can be summarized as follows:

$$J = -\sum_{i=1}^{m} y_i \log(h_\theta(x_i)) + (1 - y_i) \log(1 - h_\theta(x_i))$$

where,
m is the number of training examples,
y_i is the true class label for the i-th example (either 0 or 1),
$h_\theta(x_i)$ is the predicted probability for the i-th example, as calculated by the logistic regression model.
Therefore,
The Cost Function is given as:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & if \quad y = 1 \\ -log(1 - h_\theta(x)) & if \quad y = 0 \end{cases}$$

2.3 XGB Regression

XGBoost(eXtreme Gradient Boost) is a powerful approach for building supervised regression models. The validity of this statement can be inferred by knowing about its (XGBoost) objective function and base learners. The objective function contains loss function and a regularization term. It tells about the difference between actual values and predicted values, i.e how far the model results are from the real values. The most common loss functions in XGBoost for regression problems is reg:linear, and that for binary classification is reg:logistics. Ensemble learning involves training and combining individual models (known as base learners) to get a single prediction, and XGBoost is one of the ensemble learning methods. XGBoost expects to have the base learners which are uniformly bad at the remainder so that when all the predictions are combined, bad predictions cancels out and better one sums up to form final good predictions.

2.3.1 XGB Objective Function

XGBoost for regression uses an objective function that measures the difference between predicted and actual values. Common objective functions for regression include squared error loss, absolute error loss, or variants that penalize large errors differently. The objective function (loss function and regularization) at iteration t that we need to minimize is the following:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i})^{(t_i-1)} + f_t(x_i)) + \Omega(f_t)[4]$$

Here,

n is the number of training examples,

K is the number of trees in the ensemble,

y_i is the true target value for the i-th example,

$\hat{y_i}$ is the predicted value for the i-th example,

$f_t(x_i)$ represents the t-th tree in the ensemble.

$\mathcal{L}$ is the loss function measuring the difference between the true and predicted values,

$\Omega(f_t)$ is a regularization term penalizing the complexity of the t-th tree.

The loss function l typically used for regression tasks is the mean squared error (MSE):

$$\mathcal{L}(y_i, \hat{y_i}) = (y_i - \hat{y_i})^2$$

2.3.2 Tree Complexity

This term encourages sparsity in the leaf scores to prevent overfitting. It is defined as:

$$\Omega(f_t) = \gamma T + \lambda ||\omega||_1$$

Here,
$||\omega||_1$ is the L1 norm of the leaf scores. The objective function is minimized through a process called gradient boosting, where each new tree is trained to minimize the residual errors of the ensemble of trees built so far.

During training, the algorithm computes the gradient of the loss function with respect to the predicted values, and the new tree is fit to the negative gradient, effectively reducing the loss in the direction that would minimize the objective function.

This iterative process continues until a stopping criterion is met, such as reaching a maximum number of trees or achieving a minimal improvement in the objective function on a validation set.

In summary, XGBoost for regression optimizes an objective function that consists of a loss term measuring prediction errors and a regularization term penalizing the complexity of the trees in the ensemble. Through gradient boosting, the algorithm iteratively improves the ensemble of trees to minimize the objective function and make accurate predictions for regression tasks.

CHAPTER 3

Methodology

1. **Data Collection:** Various datasets were researched and observed for this study. However since the main focus was on comparing the performance rather than an applicable real world outcome, synthetic dataset was considered.

2. **Data Preprocessing:** In the data preprocessing phase, missing values within the dataset were carefully addressed. Through a comprehensive examination, the locations and extent of missing data were identified, and appropriate strategies for numerical data were applied.

3. **Baseline Regression Modeling:** In this phase we crafted baseline regression models using conventional techniques,excluding the integration of Principal Component Analysis (PCA). These models were developed to serve as foundational benchmarks. Linear,Logistic and XGBoost regression algorithms were applied to the preprocessed datasets. As a reference, these baseline models represented the initial state of predictive capabilities without dimensionality reduction. The result evaluation, centered on the key metric R-squared, facilitated a comprehensive comparison between baseline models and those enhanced by PCA, forming the basis for measuring the impact of PCA on model performance and interpretability.

4. **PCA Application:** The dataset went under a transformative process as Principal Component Analysis (PCA) was systematically applied. The objective was to capture the essential information while reducing dimensionality. Components were selected based on their ability to explain the variance within the dataset, with techniques which explained variance guiding the optimal number of principal

components. By retaining components that contributed significantly to the overall variance, a more concise representation of the data was achieved.

5. **Dimensionality-Reduced Regression Modelling:** In this phase of dimensionality-reduced regression modeling, a new model was trained on the dataset transformed through dimensionality reduction techniques, which is Principal Component Analysis (PCA). This model aimed to hold the essential information captured by a reduced set of features while maintaining or even improving predictive capabilities. The performance metrics for this newly trained model were systematically evaluated, employing key measures such as R-squared. This analysis allowed for a direct comparison between the baseline regression models and the dimensionality-reduced model.

6. **Resource usage:** In the evaluation of the regression models,the training time was diligently recorded. This surrounded the duration each model required for training, providing insights into the computational efficiency of the baseline and dimensionality-reduced models. While the timing aspect offered an immediate measure of efficiency gains, it is noteworthy that memory usage will be considered in our future work.

7. **Visualisation:** As part of the analysis, the graphical representation allowed for insights into the dimensionality reduction achieved by Principal Component Analysis (PCA). The graphs were generated to showcase the application of PCA to all three models, illustrating the impact of dimensionality reduction on each model.

CHAPTER 4

Applications

4.1 Data Collection

The data were imported from the site: https://statisticsglobe.com/[5], where open sources datasets for various dimensions were available. The dataset that was choosen for this study consists of 10,000 rows and 201 columns, where one column is the dependent(response) variable and other are independent(exploratory).

4.2 Output of Regression Models

In the analysis below, the yellow lines in graph refers to R-squared values whereas the blue lines correspond to execution time. Here the time refers to the time taken by the computing device to adjust suitable parameters for the model.

4.2.1 Output of Linear Regression Model

The results analyzed of the Linear regression model without using PCA is:
The average R^2 score is 0.999
The average time taken is 0.084s

Now after applying PCA to the dataset, resulting in different number of components the final results are shown in the graph below.

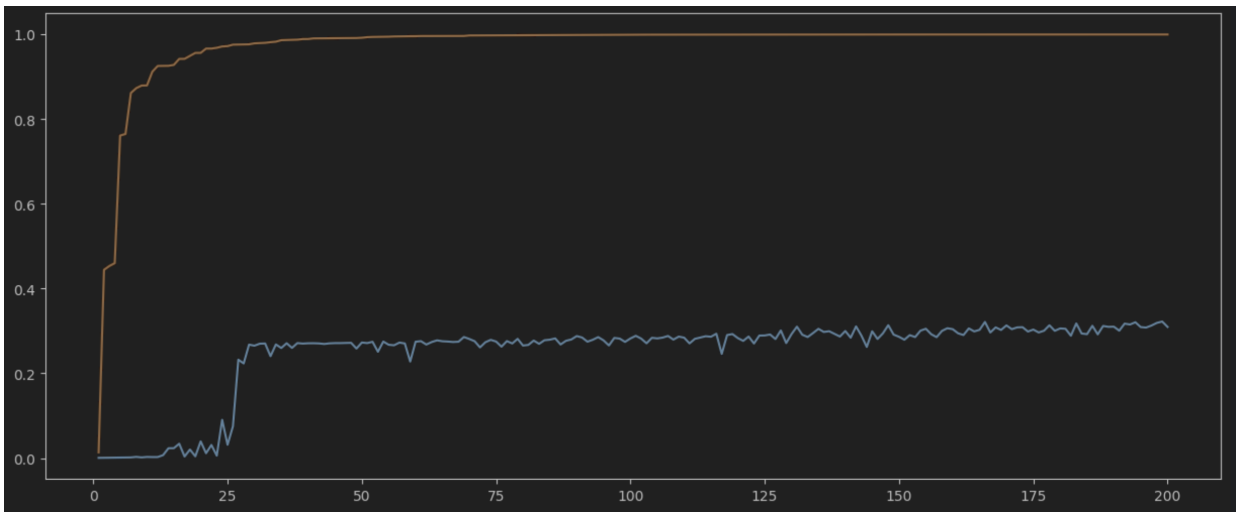

Figure 4.1: R^2 and time of Linear Regression model with variable no. of components

In examining the graph, we discern a notable pattern the R-squared values, as they exhibit a prolonged period of stability before experiencing a sudden and pronounced decline. On the other hand, the computational time, it showcases on average a gradual decrease in time.

To find the best trade-off, around 25 no. of components seems to hit the mark. Here, if a slight dip in R-squared feel reasonable i.e. some compromise works, then the number of components that can be taken is below 25. But if no compromise is wanted in the R-squared then we take no. of components above 25 along with examining the execution time. For this model, the prime range for efficiency is between 25 and 50 no. of components.

4.2.2 Output of Logistic Model

The results analyzed of the Logistic regression model without using PCA is:

The average R^2 score is 0.963

The average time taken is 0.174s

Now after applying PCA to the dataset, resulting in different number of components the final results are shown in the graph below.

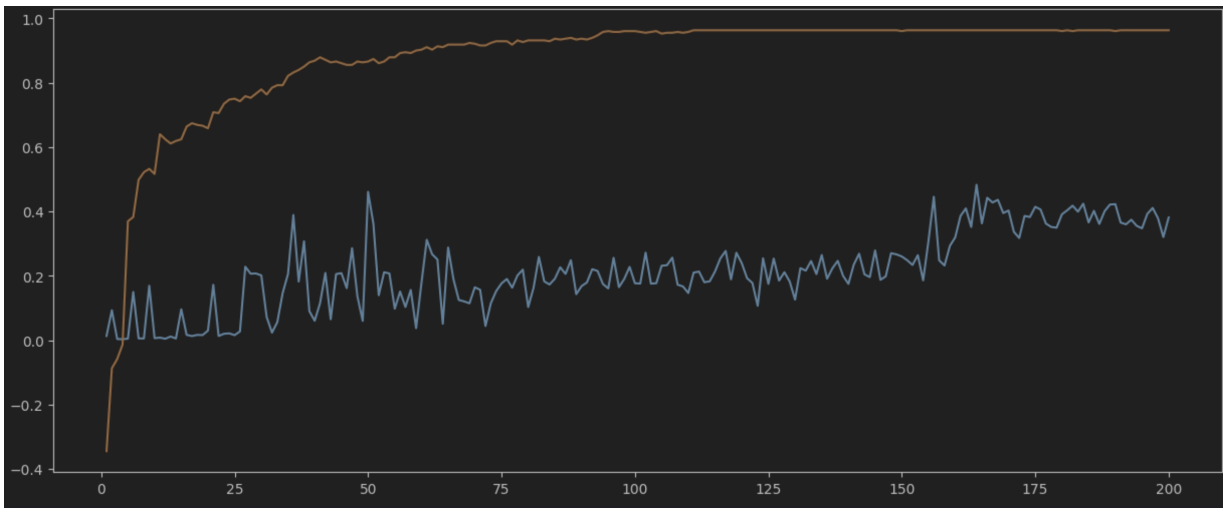

Figure 4.2: R^2 and time of Logistic Regression model with variable no. of components

Similarly, when we look at the graph of logistic model, when the no. of components is at 40 and below, it takes the lowest time but the R-squared associated with it has a huge compromise i.e is around 0.7. And thus around this region, PCA is unable to explain the variance properly. So the model is guaranteed to explain the variance accordingly. To find the best trade-off, around 50 to 100 no. of components seems fine. Here, the decrease in time and a slight dip in R-squared feel reasonable if some compromise works. But in situation of no compromises in the R-squared, 125 no. of components and above can be considered along with examination of execution time. According to the above graph, around 140 no. of components seems to be optimal for this logistic model.

4.2.3 Output of XGBoost Model

The results analyzed of the XGBRegressor model without using PCA is:
The average R^2 score is 0.954
The average time taken is 1.222s
Now after applying PCA to the dataset, resulting in different number of components the final results are shown in the graph below.

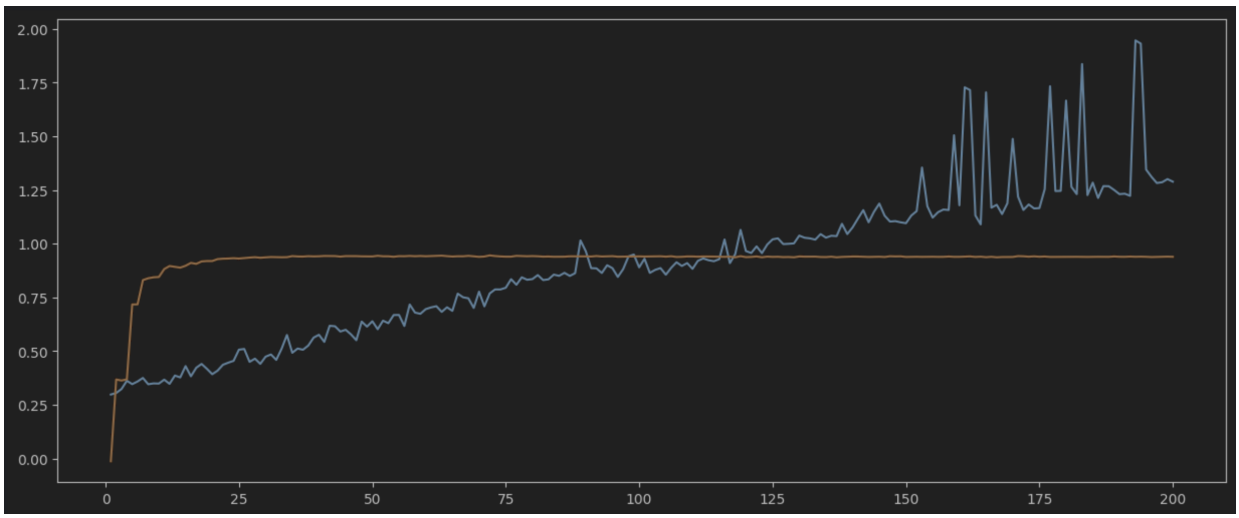

Figure 4.3: R^2 and time of XGBRegressor model with variable no. of components

While observing the graph of XGBRgegressor, we can observe that the R-squared score remains almost the same from 25 no. of components to 200. But then decrease rapidly if we try and go below 25. Now when we observe the computational time, the trend is decreasing agreeably. In comparison to other regression models above, XGBRegressor is a time consuming model. We observe that the time has gone up to a maximum of 1.9 seconds on average in particular cases.

When we look at around 15 no. of components, the model takes the lowest time. However since there is a significant dip in the R-squared, the decrease in time is only applicable if the compromise in R-squared in reasonable. But if no compromises in R-squared is wanted, no. of components chosen are 25 and above along with the examination of computation time. So around 25 no. of components seems to be the optimal no. of components for this XGBRegressor model.

The main challenge of PCA is that it is not necessary that the time decreases with decrease in number of components as shown in the graphs above, the time taken and number of components may be positively correlated but are not directly proportional. So to take this factor into account, manual inspection is recommended.

4.3 Some applications

1. Reducing the size of images:

PCA can be used to reduce the size of an image without significantly impacting the quality. Beyond just reducing the size, this is useful for image classification algorithms.

2. Medical diagnosis and disease prediction:

Healthcare datasets often contain a multitude of features related to patient health. Applying regression models for medical diagnosis, such as predicting disease progression or assessing the effectiveness of treatments, can be challenging due to the high-dimensional nature of the data. By integrating PCA, you can uncover the most critical factors influencing patient outcomes, potentially improving model interpretability.

3. Finding patterns in high-dimensional datasets:

Examining the relationships between principal components and original features can help uncover patterns in the data that are harder to identify in our full dataset.

4. Stock price prediction

Many models of stock price prediction rely on estimating covariance matrices. However, this can be difficult with high-dimensional data. PCA can be used for data reduction to help solve this issue.

5. Anomaly detection in CyberSecurity:

Cybersecurity involves analyzing large datasets to detect abnormal patterns indicative of potential security threats.Regression models can be used for anomaly detection, but the high dimensionality of cybersecurity data can hinder model performance. By applying PCA, your study can assess the effectiveness of regression models in identifying and predicting cybersecurity anomalies.

4.4 Conclusion

In our exploration into the influence of Principal Component Analysis (PCA) on various regression models, our study has uncovered valuable insights into the delicate balance between computational efficiency and model performance. The analysis of Linear, Logistic, and XGBoost regression models using PCA has shown the trade-offs that researchers and practitioners must carefully consider. The findings emphasize the importance of manual inspection and thoughtful consideration in choosing the optimal number of components for a given regression model. The non-linear relationships observed, especially in XGBoost, underscore the complexity of balancing time efficiency and model accuracy.

While PCA presents a powerful dimensionality reduction tool, its application requires a tailored approach, taking into account the specific characteristics of the regression model in use. As the landscape of data-driven decision-making continues to evolve, our findings contribute valuable insights for practitioners seeking to harness the potential benefits of

PCA in real-world applications.

REFERENCES

[1] J. De Leeuw, "History of nonlinear principal component analysis," *Visualization and Verbalization of Data*, vol. 751, 2011.

[2] J. L. Lastovicka and J. E. Jackson, "A user's guide to principal components." *The Statistician*, vol. 42, pp. 76–77, 1991. [Online]. Available: https://api.semanticscholar.org/CorpusID:50436960

[3] [Online]. Available: https://en.wikipedia.org/wiki/Principal_component_analysis

[4] D. Leventis, "A walkthrough of the gradient boosted trees algorithm's maths," 2018.

[5] [Online]. Available: https://statisticsglobe.com/datasets-for-pca