Text

Description automatically generated with medium confidence

**Introduction**

**What is PCA:**

Principal Component Analysis (PCA) is an incredibly useful and widely used multivariate technique in Machine Learning. Moreover, such technique is also extremely helpful in the analysis of huge datasets, whilst effectively undertaking Dimensionality Reduction and Feature Selection. In continuation, PCA is used to ensure that data scientists would be able to load and use large datasets on less powerful machines, which could not support the size of the full dataset. Additionally, PCA also provides cleaner data visualisation through the envisioning of the key data features in the full dataset, which hold the largest degree of information. [1-2]

**History:**

PCA has a long and illustrious history that goes back more than a century. The algorithm was pioneered by Karl Pearson, who in 1901 launched this system with the aim of undertaking data analysis and dimensionality reduction. The current PCA's design was first provided by Harold Hotelling in the 1930s, but it wasn't until then that the method truly started to take shape.

Hotelling was instrumental in formulating the concept of variance maximization and the use of orthogonal projections to find the Principal Components. Further improvements to the PCA algorithm were developed in the 1960s, in part due to the emergence of Singular Value Decomposition (SVD), which offered an alternate method for calculating the eigenvalues and vectors necessary to do the PCA. The growing adoption of PCA at this time was largely compelled by the need for dimensionality reduction and the widespread growth of computers. The method gained a lot of traction in the 1970s and later when data scientists and researchers comprehended how effective it was at dealing with the enormous, complex datasets that were becoming more and more prevalent in industries such as banking, engineering, and medicine. [1-2]

**Motivation and Objectives:**

Nevertheless, such algorithm’s behaviour may not always be comprehensible, thus cementing the need for the creation of a visual tool, which could allow users to visualise the algorithm's stages and data transformations, whilst offering a better understanding on the modified data. Consequently, the programmed solution also effectively portrays the PCA process as a simple convenient story targeted towards students who have just completed a Linear Algebra or AI Numerical Methods course. The created Jupyter Notebook which outlines the aforementioned process conveys to the students, the necessary information to be able understand such algorithm, whilst providing the students with essential tools to be able to experiment and expand their knowledge. Additionally, the notebook was also designed to be robust and responsive, allowing students to interact with the visual plots through the plot’s minimising and maximising tools. Accompanying the Jupyter Notebook, the developed solution comes equipped with various famous datasets utilised by the machine learning community such as the Iris dataset, with the aim to make the students familiar with such datasets. In addition, the datasets were chosen for their distinct properties, to allow students to evaluate different experiments and infer new knowledge.

**Summary of Results:**

The main results obtained from the creation of such artefact pertains to the following list:

1. Principle Component Analysis is designed to be utilised on linearly separable data i.e., the variables in the dataset need to be linked together through a linear relationship. Consequently, Kernel PCA can be used to resolve such issue [3].
2. The PCA algorithm is intended to be used on continuous data values. The approach taken to cater for discrete value features in the dataset, included changing the discrete values to continuous values, through different encoding techniques. Moreover, such discrete columns could also be discarded before the calculation of the PCA, which would result in more accurate data projections, however incurring the loss of the discrete data columns.
3. There are different Libraries which may be used to implement PCA, for example the NumPy and the scikit-learn library. Furthermore, sometimes the NumPy library does not support the singular value decomposition of large datasets, and would sometimes crash, whilst, the scikit-learn library does not support such issue, as it utilises random singular value decomposition. Randomized SVD can approximate the whole SVD with a substantially lower computation cost by randomly selecting a fraction of the matrix's rows or columns [4].
4. It is imperative that the data which is fed to the PCA algorithm is normalized, since if given unnormalized data (some data will have a high variance, and some will have a low variance), PCA will load on the high variance data [5]. Additionally, the created artefact utilises Z-Score Normalization.
5. Understanding the PCA algorithm mathematically can be quite a difficult task, as the process includes multiple mathematical calculations such as SVD or Covariance Matrix evaluation. Nevertheless, the erected artefact provides the need-to-know basis for such algorithm, in a well explained format.

**Background and Methodology**

Mathematically, PCA enables the conversion of linear continuous data into a new coordinate system, characterized by new axis (Principal Components) which are ordered in accordance with the features in the new coordinate system. Continuously, the best principal components can be plotted on different dimensional graphs, in order to provide a satisfactory visualisation of a large dataset. Unfortunately, such method may have some minimal data reduction, however visualising an n dimensional feature dataset on a 3D plot is quite a benefit. The PCA's main characteristics of decreasing the dimensionality of data, whilst retaining salient information, lead to it being the most effectively ranked data analysis and machine learning technique [1-2].

The designed implementation explains the process of the Principal Component Analysis, in the following ordered sections:

[**1. Loading the Data**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#loadData)

A key step before the initiation of the PCA Algorithm involves the selection of a relevant dataset which will be analysed by such algorithm. The designed implementation provides students interacting with the notebook to choose any of the default datasets and explore how the PCA algorithm will function on such datasets. Additionally, students are also given the option to load their preferred dataset. Moreover, students are also highly encouraged, before commencing the Principal Component Analysis, to thoroughly analyse and understand the dataset's properties and qualities. In addition, the default datasets chosen, were selected with the purpose of having different attributes. This was done, in order to allow students to test out different experiments and be able to compare the result obtained through varying the dataset.

**The following are the default datasets (Obtained from [6-12]):**

1. **country\_wise\_latest.csv** - This dataset has a small Size, a large number of Features, and a few numbers of Discrete Columns.
2. **diabetes.csv** - This dataset has a small Size, a small number of Features, and no Discrete Columns.
3. **FIFA - 2014.csv** - This dataset has a small Size, a small number of Features, and one Discrete Column.
4. **IRIS.csv** - This dataset has a small Size, a large number of Features, and one Discrete Column.
5. **Salary\_Dataset\_with\_Extra\_Features.csv** - This dataset has a large Size, a small number of Features, and a reasonable number of Discrete Columns.
6. **spotify.csv**- This dataset has a large Size, a large number of Features, and a reasonable number of Discrete Columns.
7. **wine-quality-white-and-red.csv**- This dataset has a large Size, a large number of Features, and one Discrete Column.

[**2. Dataset Feature Selection**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#featureSelection)

Another key step when performing a data analysis or a machine learning study, pertains to observing the type and number of different **Genes/Features**, which the dataset has. Furthermore, this step is relatively important, as sometimes processing a huge number of features in the dataset may cause memory allocation issues or prolong the processing time of algorithms.

**---Please note that in case less than three columns are chosen, the first three columns will be added to the filtered dataset. This is done, to ensure that the filtered dataset, would have enough features for visualisation in the upcoming sections.------**

[**3. Dealing with Discrete Data**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#discreteData)

As previously mentioned, PCA is designed to be utilised on continuous data [13]. Thus, placing the need to transform discrete data into continuous data before using PCA on a dataset. This is done, as discrete data lacks a continuous range of values and cannot be represented in the same way as continuous data for this cause.

There are various ways how discrete data can be Transformed/Encoded to continuous data, in order to be examined by the PCA.

**The following are different types of Encoders, which were implemented:**

1. **One-Hot Encoding**
2. **Label Encoding**
3. **Ordinal Encoding (Similar to Label Encoding)**
4. **Count Encoding**
5. **Word Embeddings Model**

**What is One-Hot Encoding?**  
One-hot Encoding is a data preparation technique used to transform discrete variables into a format that machine learning algorithms can examine. Consequently, this encoding algorithm works by creating a binary vector for each possible category in the data. Additionally, each binary vector would have a value of 1 or 0 to indicate the presence or absence of each category respectively. [14-16]

-----In the code cell below, One-Hot Encoding is applied to the first Discrete Data Column, via the pd.get\_dummies function.-----

Additionally, such encoding is quite explosive, as the number of different Genes/Features obtained after applying One-Hot encoding on a single column, will greatly increase the number of columns depending on the number of distinct features in each column. For an algorithm which aims to reduce dimensionality, such approach to turn discrete data into continuous data is quite inefficient, notwithstanding the increase in memory and time complexity presented.

Continuously, one might think whether this binary vector can be transformed back to decimal. Note that such encoding algorithm exists and is known as Binary to Decimal Decoding. The aforementioned algorithm effectively transforms the binary vector back into a decimal value, thus reducing the size of the Genes/Features to their original number [17]. Essentially such encoding would take relatively more time whilst achieving the same results as Label Encoding or Ordinal Encoding.

**What is Label Encoding?**  
Label Encoding is another data preparation technique used to transform discrete variables into a format that machine learning algorithms can examine. Such encoder works by giving each distinct category a unique numeric value or code [14,15,18]. For instance, taking the list of categories ["hat","apple","cap"] will be encoded as [3,1,2] (as numeric values).

------In the code cell below, Label Encoding is applied through the pd.factorise function, and the sort flag applied to True, so that there wouldn't be in the order which they appeared first.----

**What is Ordinal Encoding?**  
Ordinal Encoding is another data preparation technique used to transform discrete variables into a format that machine learning algorithms can examine, which works similarly to Label Encoding. Such encoder works by giving each distinct category a unique numeric value or code, based on the order which the category appeared first [14,15,19]. For instance, taking the list of categories ["hat","apple","cap"] will be encoded as [1,2,3] (as numeric values, and encoded in the order which they appeared).

-------------------In the code cell below, Ordinal Encoding is applied through the pd.factorise function, and the sort flag applied to False, so that the elements would be classified in the order which they appeared first.----------

**What is Count Encoding?**  
Count Encoding is another data preparation technique used to transform discrete variables into a format that machine learning algorithms can examine. Such encoder works by encoding each distinct category, with the number of times such category appeared [14-16]. For instance, if the category "hat" appeared 5 times, then "hat" will be encoded by the number 5.

-------In the code cell below, Count Encoding is applied through the .value\_counts and .map function on the discrete data.--------

**What is a Word Embeddings Model?**  
A Word Embeddings Model is a sort of natural language processing (NLP) model which depict words as numerical vectors in a high-dimensional space. This model works by first training a neural network on a large corpus of text data, in order to represent words as dense, low-dimensional vectors. Each component of the word vector represents a specific aspect or characteristic of the word, such as its semantic meaning, part of speech, or syntactic context [20]. Moreover, the developed artefact focuses on the use of Word2vec, which is a type of Word Embeddings Model [20].

----In the code cell below, the Word Embeddings Model (Word2vec) is applied on the discrete data. Please note that the process may take some time to complete.------

Consequently, all of the aforementioned encoding techniques were designed, with the purpose to familiarise students with different encoding techniques. Moreover, the large stack of encoding techniques also provides students with the liberty to test out the different techniques on different datasets, as it could result that a particular technique would outperform the others for a specific dataset.

[**4. Filtered Dataset Visualisations**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#initialDataSetVisualisations)

Visualisation is a useful tool, as it aids in the process of identifying visual patterns and characteristics in data. Subsequently, it is simpler for individuals to spot patterns and trends when data is represented visually than when it is presented in numerical or written form. Unfortunately, not all features can be visualised, as visualised data is limited to three dimensions, thus individuals need to choose which features to visualise, from a high-dimensional dataset with many features.

-------In the code cell below, Students are presented with the list of features in the dataset, and are given the option to choose either feature for the three Dimensional Variables, which will be Represented Visually.----------

[**5. Normalizing Data**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#normalizingData)

Normalization is the process of converting and scaling the numerical characteristics inside a dataset to enable the data to have a uniform range and distribution. Furthermore, normalization's primary objective is to guarantee that no feature dominates or has an excessively large impact on the model's performance. This process is key, in the calculation of the PCA, since if given unnormalized data, the PCA algorithm will load on the high variance data [5]. An example would be having two data variables, one having a value of 1 and the other having a value of 700, whereby the PCA algorithm will issue higher importance to the second value. Recalling the previous encoding techniques, encoding techniques such as Label Encoding or Ordinal Encoding will provide transformed variables with an uneven distribution, i.e., some values will have a large integer value and others will have a small integer value. Through normalization the aforementioned issues can be resolved, and the data values would all be converted into a uniform range.

The developed implementation focuses on utilising **Z-Score Normalization** or also known as **Standardization** [21], and such normalization technique can be constructed through the following formula:

A picture containing text

Description automatically generated

### Additionally, the following is the formula used to calculate the Standard Deviation (σ) :

A picture containing diagram

Description automatically generated

[**6. Normalized Dataset Visualisations**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#normalizedDataSetVisualisations)

Through comparisons with the original dataset visualisation and the normalized data visualisation, one might note that the normalized data is plotted on a smaller range of values, when compared to the original plots in the section above. Additionally, one can also notice how in the normalized plots, the data values are centred around zero.

------------Please also note that the colour of the points in this graph may change slightly from the above graphs, as the data values are now centred around zero and the normalized plot is plotted on a smaller range of values.------------

[**7. Understanding PCA - SVD Approach**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#pcaSVD)

Singular Value Decomposition (SVD) is a decomposition method which is utilised to factorise a matrix of **m x n** size into three components. The resultant components include **U** and 𝑉T, which are two orthonormal matrices, and Σ which is a diagonal matrix containing the singular values of the original matrix. Additionally, the size/magnitude of each singular value signifies the importance in explaining the data [22]. For example, a singular value of 10 will have a higher importance than a singular value of 5.

---------Taking a small subset of the entire dataset if dataset has a larger size than a respective threshold, and working out the PCA algorithm, via the SVD Approach.

Note that the dataset is being reduced to a tenth of its size, whilst maintaining the number of columns, in order to aid the student to better understand the concept, and method of calculation, in case the dataset has a larger size than the respective threshold of 10000.---------

Text

Description automatically generated

The second step in the Calculation of PCA via SVD Approach, involves multiplying the U matrix by the Σ matrix.

This is done as the multiplication of 𝑈. Σ presents a matrix whose columns give the projections of the data points on each principal axis.

[**8. Understanding PCA - Covariance Matrix Approach**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#pcaCovariance)

A Covariance Matrix or also known as the **Covariance Variance Matrix** and is a **n x n** symmetric matrix which is used to show the covariance values between adjacent pairs of items in a dataset of n attributes. Additionally in this matrix, the diagonal elements represent the variance of each element. [23]

Table

Description automatically generated

**Calculating the PCA by multiplying the Normalised Dataframe with Sorted Eigenvectors**

This is done as the multiplication of Normalized Dataframe. Eigenvectors presents a matrix whose columns give the projections of the data points on each principal axis.

[**A**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#pcaCovariance) **note on Calculating the Variance Ratio and Visualising ratio in Scree Plot**

The use of variance ratios in PCA is done, in order to calculate the percentage of the overall variance in the data that each principal component contributes to. Each principal component in the PCA algorithm captures a specific amount of data variation, thus we can determine the percentage of the overall variation that each component accounts for by computing the variance ratio. Additionally, through the variance ratio we are able to determine which principal components are crucial for explaining the variation in the data. [24]

**The Variance Ratio is calculated through the following Formula (obtained from [24-25]):**

Text

Description automatically generated with medium confidence

Additionally, through the use of the Scree Plot, which displays the percentage of variation explained by each primary component, one is able to determine the number of components required to account for a specific percentage of the overall variance in the data [26].

The following methods can be used to determine the optimal number of principal components to retain [26]:

1. **Elbow Method** - This method of selection adopts to retain all the principal components prior to the curve plateau in the Scree Plot. Moreover, this method works by pinpointing the point on the Scree Plot where the curve plateaus, and then selecting the number of components before this point as the ideal number of components to maintain.
2. **Kaiser Rule** - This method of selection selects to retain all the principal components with eigenvalues which have at least a value of 1.
3. **Proportion of Variance Plot** - This method of selection chooses to retain all the principal components which represent a percentage (%) amount of the variance.

[**9. Comparisons Between Approaches**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#approachCompare)

[**10. Working out PCA on the Entire Dataset**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#pcaEntire)

[**11. PCA Visualisations**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#pcaVisualisations)

[**12. Conclusions and Limitations of PCA**](http://localhost:8888/notebooks/IAPT/A%20Guide%20to%20Principal%20Component%20Analysis.ipynb#conclusion)

Moreover, there are two approaches how the PCA algorithm can be calculated: The SVD approach and the Covariance Matrix Approach.

**Methodology**

**Evaluation**

**Conclusion**

**References**

[1] S. Mishra et al., "Multivariate Statistical Data Analysis-Principal Component Analysis," Int. J. Livest. Res., vol. 1, pp. 1-6, 2017. [Online]. Available: https://www.researchgate.net/publication/316652806\_Principal\_Component\_Analysis. [Accessed: 18-Apr-2023].

[2] D. Li and S. Liu, "4.2.3.1 Principal Component Analysis," in Water Quality Monitoring and Management: Basis, Technology and Case Studies, 1st ed., S. K. Gupta and R. Kumar, Eds. Amsterdam, Netherlands: Elsevier, 2019. [Online]. Available: https://www.sciencedirect.com/topics/agricultural-and-biological-sciences/principal-component-analysis. [Accessed: 18-Apr-2023].

[3] N. B. Subramanian, "Types of PCA", aiaspirant.com. [Online]. Available: <https://aiaspirant.com/types-of-pca/>. [Accessed: 18-Apr-2023].

[4] N. Halko, P. G. Martinsson, and J. A. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions,” arXiv preprint arXiv:0909.4061, 2009. [Online]. Available: <https://arxiv.org/abs/0909.4061>. [Accessed: 18-Apr-2023].

[5] Stack Exchange. "Why do we need to normalize data before Principal Component Analysis (PCA)?", Cross Validated, May 26, 2014. [Online]. Available: <https://stats.stackexchange.com/questions/69157/why-do-we-need-to-normalize-data-before-principal-component-analysis-pca>. [Accessed: 18-Apr-2023].

[6] DEVAKUMAR K. P., "COVID-19 Dataset", Kaggle, 2020. [Online]. Available: <https://www.kaggle.com/datasets/imdevskp/corona-virus-report?select=country_wise_latest.csv>. [Accessed: 18-Apr-2023].

[7] UCI MACHINE LEARNING, "Pima Indians Diabetes Database", Kaggle, 2016. [Online]. Available: <https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database>. [Accessed: 18-Apr-2023].

[8] S. BANERJEE, "FIFA - Football World Cup Dataset", Kaggle, 2022. [Online]. Available: <https://www.kaggle.com/datasets/iamsouravbanerjee/fifa-football-world-cup-dataset?select=FIFA+-+2014.csv>. [Accessed: 18-Apr-2023].

[9] MATHNERD, "Iris Flower Dataset", Kaggle, 2018. [Online]. Available: <https://www.kaggle.com/datasets/arshid/iris-flower-dataset>. [Accessed: 18-Apr-2023].

[10] S. BANERJEE, "Software Industry Salary Dataset - 2022", Kaggle, 2022. [Online]. Available: <https://www.kaggle.com/datasets/iamsouravbanerjee/software-professional-salaries-2022>. [Accessed: 18-Apr-2023].

[11] R. Holbrook and A. Cook, "Principal Component Analysis, spotify.csv", Kaggle. [Online]. Available: <https://www.kaggle.com/code/ryanholbrook/principal-component-analysis/data?select=spotify.csv>. [Accessed: 18-Apr-2023].

[12] RUTHGN, "Wine Quality Data Set (Red & White Wine)", Kaggle, 2022. [Online]. Available: <https://www.kaggle.com/datasets/ruthgn/wine-quality-data-set-red-white-wine>. [Accessed: 18-Apr-2023].

[13] V. Karthik, "PCA for categorical features", Stack Overflow, Dec. 2016. [Online]. Available: <https://stackoverflow.com/questions/40795141/pca-for-categorical-features#:~:text=PCA%20is%20designed%20for%20continuous,yes%2C%20you%20can%20use%20PCA>. [Accessed: 18-Apr-2023].

[14] Datagy. "Pandas get\_dummies (One-Hot Encoding) Explained," Datagy.io, Feb. 2021. [Online]. Available: <https://datagy.io/pandas-get-dummies/>. [Accessed: 18-Apr-2023].

[15] DataCamp. "Dealing with Categorical Data". DataCamp, 2021. [Online]. Available: <https://www.datacamp.com/tutorial/categorical-data>. [Accessed: 18-Apr-2023].

[16] B. Roy, "All about Categorical Variable Encoding," Towards Data Science, Jul. 2, 2019. [Online]. Available: <https://towardsdatascience.com/all-about-categorical-variable-encoding-305f3361fd02>. [Accessed: 18-Apr-2023].

[17] T. Crosley, "What is the binary to decimal decoder?", Quora, May 8, 2018. [Online]. Available: <https://www.quora.com/What-is-the-binary-to-decimal-decoder>. [Accessed: 18-Apr-2023].

[18] Pandas. "pandas.factorize()". pandas 1.4.0 documentation, Jan. 07, 2022. [Online]. Available: <https://pandas.pydata.org/docs/reference/api/pandas.factorize.html>. [Accessed: 18-Apr-2023].

[19] J. Brownlee, "One-Hot Encoding for Categorical Data," Machine Learning Mastery, Aug. 17, 2020. [Online]. Available: <https://machinelearningmastery.com/one-hot-encoding-for-categorical-data/>. [Accessed: 18-Apr-2023].

[20] Vatsal, "Word2Vec Explained", Towards Data Science, Jul. 29, 2021. [Online]. Available: <https://towardsdatascience.com/word2vec-explained-49c52b4ccb71>. [Accessed: 18-Apr-2023].

[21] R. Sharma. "What is Normalization in Data Mining and How to Do It?", UpGrad, Sep. 22, 2022. [Online]. Available: <https://www.upgrad.com/blog/normalization-in-data-mining/#:~:text=Project%20Ideas%20%26%20Topics-,Z%2DScore%20Normalization,up%20to%20%2B3%20standard%20deviation>. [Accessed: 18-Apr-2023].

[22] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular Value Decomposition and Principal Component Analysis," in Learning from Data: Concepts, Theory, and Methods, vol. 2, Springer, Boston, MA, 2007, pp. 151-176, doi: 10.1007/0-306-47815-3\_5. [Online]. Available: <https://www.researchgate.net/publication/2167923_Singular_Value_Decomposition_and_Principal_Component_Analysis>. [Accessed: 18-Apr-2023].

[23] CUEMATH, "Covariance Matrix", CUEMATH. [Online]. Available: <https://www.cuemath.com/algebra/covariance-matrix/>. [Accessed: 18-Apr-2023].

[24] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," in The Data Deluge: Can Libraries Cope with E-Science? Proceedings of a Conference Held at the Royal Society, London, UK, 4-5 November 2004, vol. 463, Royal Society Publishing, 2016, pp. 21-36. doi: 10.1098/rsta.2015.0202.[Online]. Available: <https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202>. [Accessed: 18-Apr-2023].

[25] K. Guillaumier, "Linear Algebra in Data Science and PCA"

[26] S. Mangale, "Scree Plot," Medium, Aug. 28, 2020. [Online]. Available: <https://sanchitamangale12.medium.com/scree-plot-733ed72c8608>. [Accessed: 18-Apr-2023].